2 Cognitive Systems Engineering at 20-Something: Where Do We Stand?

ROBERT G. EGGLESTON
U.S. Air Force Research Laboratory

ABSTRACT

Cognitive Systems Engineering (CSE) is a multidisciplinary field that is both a scientific endeavor and an engineering practice that emphasizes user-centered analysis and design. This chapter presents an analysis of the state of CSE that concentrates on its technical development. The analysis considers the field from three different perspectives: conceptual foundations, engineering tools and practices, and deployment and use in systems design. Four distinct cognitive engineering frameworks that span the field are identified and discussed as a way to make clear similarities and differences across the conceptual landscape that serves to define the field. The CSE "toolkit" is briefly discussed in terms of knowledge elicitation, knowledge capture, and design innovation support. Some areas of current confusion and ambiguity are also presented as a means to highlight possible impediments to continued growth and technical maturity of the field. The chapter concludes with comments about the apparent direction of CSE developments as they relate to large-scale system design.

2.1 INTRODUCTION

Cognitive Systems Engineering (CSE) is both a field of scientific study and an approach to human-centered engineering practice for the design of interactive systems. While it has been defined in many different ways by various authors, at its core CSE seeks to understand how to model work in ways *directly useful for design* of interactive systems. Further, it seeks to develop concepts, principles, tools, and methods that enable work-centered design engineering of interactive systems. As an engineering practice, CSE exploits its scientific understanding and principle-based engineering toolkit to produce design products

that feed into the full interactive systems development process. Products range from models of interaction, at several levels of detail, to quantitative estimates of performance for design configurations, to the innovation of support systems derived from the CSE body of knowledge.

This chapter is a commentary on the field of CSE[1]. It evolved from a discussion on the "maturity" of the CSE field that I led at the TP7 Workshop on CSE. The purpose of the commentary is to consider the state of development of CSE as a cross-disciplinary field of study and practice.

CSE has followed a logical progression in its development. In the early years, major attention was directed toward establishing the conceptual foundations of the field. Leaders of the field concentrated first on establishing the constructs used to characterize work and second, on the concepts for how to utilize these characterizations to model work in ways that would be directly useful for the practice of engineering design. This conceptual foundations period, which occurred largely in the 1980s and earlier, was followed by a time when the focus of attention shifted to extending models of work and creating new data collection and analysis tools that made more explicit the cognitive aspects of work as practiced. Utilization of these knowledge elicitation and cognitive task analysis processes represents a major part of CSE as an engineering practice. This engineering practice period has dominated the 1990s. As we embark on the 21st Century, CSE seems poised to move into a third period or stage. There are early signs that the new focus will concentrate on how to improve the integration of CSE practices and products into systems engineering as an integral part of systems analysis, management, and design engineering practices for large-scale, interactive systems development.

In an effort to understand the state of CSE today, we will explore these three aspects of CSE development. The first task is to lay out the conceptual landscape of the field. Toward this end, I have identified four unique genotypes of CSE that represent its foundation and that also can provide a conceptual understanding of the field for the reader who has little prior background in this area. While more genotypes could be discussed in detail, the selected set is sufficient to span the conceptual landscape, and therefore reveal important similarities and differences among the various approaches being adopted by practitioners in the field. Thus, a review of these genotypes can be used both to add structure to an otherwise confusing landscape and to denote progress and maturity of the field in different ways. The discussion starts with a historical overview to uncover the reasons behind each genotype and then concentrates on detailing the conceptual distinctions fostered by these frameworks.

To be useful as engineering tools, the various CSE frameworks must be populated with information germane to a specific work domain in the context of a specific systems development project. With such information, CSE is able to support decision-making throughout the analysis, design, and evaluation processes of systems development. The second part of the chapter considers the

1. Both the terms Cognitive Engineering (CE) and cognitive systems engineering have been used in the literature to identify this approach to human factors engineering and engineering psychology. Hence, these terms may be treated as synonyms. Sometimes cognitive systems engineering is identified with a particular approach to CE, and this can add confusion. In this chapter I will use the term cognitive systems engineering to identify the field.

development of CSE from this engineering practice perspective. It begins with a look at each CSE genotype in the context of performing a cognitive task analysis in support of systems engineering. The discussion continues by looking at the more direct design and evaluation support aspects of CSE. As part of this presentation, CSE development is considered from two perspectives. The first perspective considers progress in terms of the innovation of new concepts and tools offered by CSE to improve systems engineering for more cognitively intensive human-machine systems. The second perspective considers progress in terms of the deployment or use of CSE concepts and engineering tools by CSE developers and practitioners. If the concepts and engineering tools are well formed, then it follows logically that they should be used with a higher degree of reliability. If the constructs are less mature or poorly communicated, then we can expect inconsistencies in deployment due to lack of clarity of concepts, limitations in supporting tools, or limitations in educating users about important distinctions. These issues are briefly considered in the discussion mainly by way of a few examples of CSE concepts from the different genotypes.

In summary, the commentary answers the question, *Cognitive systems engineering at twenty-something—where do we stand?* in three interrelated ways. As shown in Figure 2.1, it reviews the progression of major conceptual distinctions that serve to define the conceptual forms and theoretical maturity of the field. Second, it identifies recent additions to the CSE engineering tool kit that are motivated by these conceptual distinctions to see the maturity of the engineering processes. Finally, it looks at how CSE has been deployed as a means to see its maturity in terms of extent of use and clarity of application of CSE methods in systems development.[2]

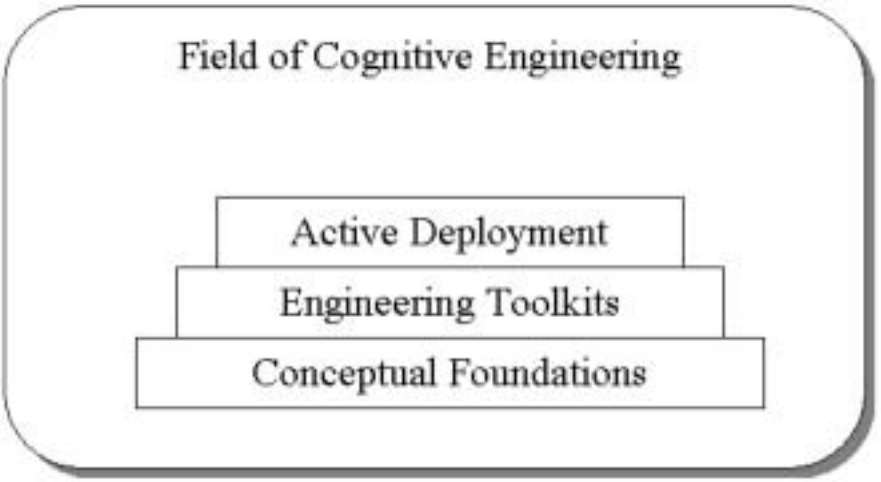

Figure 2.1: Three aspects of the state of development of cognitive systems engineering.

2. Although this commentary reviews several developments in CSE, the chapter is not intended to be a comprehensive review and survey of the field. Items have been selected to show the progression of development and the current state of CSE in terms of the three identified aspects.

2.2 WHY COGNITIVE SYSTEMS ENGINEERING?

CSE is what I do. This sentence is a paraphrase of how Donald Norman, who coined the term *cognitive systems engineering*, chose to introduce this new field of study and engineering practice (Norman, 1981, 1986, 1987). It reflects the fact that cognitive systems engineering was born as an intensely personal endeavor. Norman recognized that the movement of computer technology from the data processing center to the desktop was a harbinger of the change in work that could be expected for a large segment of the work force. In the future (i.e., today!), humans would spend much more time doing cognitive work (e.g., planning, analyzing, problem solving, decision making, negotiating, and coordinating), and computer-enhanced tools would accomplish much of the physical work (e.g., crunching numbers, assembling components, moving items, etc.) under supervisory control. Trained as both an electrical engineer and a cognitive psychologist, Norman had a strong interest in influencing the design of these new computer tools from the perspective of human performance. But the existing database regarding human performance in terms of cognitive factors, derived from engineering psychology, human factors engineering, computer science, and cognitive science, did not seem adequate. In his view, new tools were needed that concentrated at once on the fundamental understanding of cognitive work and on user-centered engineering of work support tools. This new field, CSE is thus a composite of science and engineering. It places an emphasis on a theoretical understanding of cognitive work, as well as on engineering design innovation and systemization to produce interactive systems that better support work in all of its dimensions. As Norman put it, it is the field of *applied cognitive science* (Norman, 1986).

Norman was not alone in his observation that new thinking and tools were needed to enable safe and productive work in the computer age. At about the same time David Woods and his colleagues, working in the power generation industry, called for a cognitively oriented engineering design practice to incorporate machine-based decision support in the control room (Hollnagel & Woods, 1983; Woods, 1986). Incidents like the Three Mile Island nuclear disaster heightened world awareness of the importance for good design in the operator control room of large, complex, and potentially dangerous production processes. The Woods group became aware firsthand of how the computer served to increase the cognitive distance between the operator and the details of process control. Further, this awareness was not limited to scientists and engineers in the United States. In fact, Jens Rasmussen, working quietly at the Danish Atomic Energy Commission Research Establishment (RISO), foresaw the need for a CSE long before the term was coined (Rasmussen, 1976). Rasmussen's work dates to the late 1960s, but in general was not published in widely available open source literature until the 1980s. He saw the problem in terms of a need to support adaptive problem-solving behavior as a means to

ensure safe operation of a power production facility. To him, currently accepted approaches to reliability engineering at the time did not adequately address the factors, cognitive and otherwise, that enable an operator to make adaptive and effective adjustments to meet evolving contingencies. Also in the 1980s, Stewart Card was motivated to attempt to convert advances made in the scientific studies of cognitive psychology and artificial intelligence into engineering models of human behavior that more adequately represented cognitive factors. Existing analysis tools used by human factors practitioners tended to gloss over these mental operations. As a result, cognitive characteristics of work tended to be under-represented in the analysis and design of complex person-machine systems. Card attempted to redress this problem by explicitly including cognitive variables in information-processing models of human performance.

It is evident from the foregoing that many individuals involved in the engineering research and development community were converging in the 1980s on the same conclusion: *There is a growing need to incorporate a deeper understanding of cognitive behavior into the design of modern work tools and systems.* I take this convergence as marking the formation of CSE as a new field even though, as noted, some lines of work extend back further in time. While not inclusive of all research that could rightly be included in this incipient field, these four lines of work (Norman, Woods, Rasmussen, and Card) certainly deserve to be considered as members of the central core of the CSE movement.

There are at least three important common characteristics across these four research and development programs. All of them were driven by the need to solve real-world work problems that required an effective interface between humans and machines or large machine-based processes. They all recognized the need to ensure that technology is designed to *aid the user in solving problems in work.* Finally, they all saw the need for CSE as both an applied science of joint cognitive systems and an engineering methodology or practice for the design of such systems. Thus, CSE would need to mature as a science and as an engineering practice to meet the expectations of its founders. Indeed, this has been the case.

CSE is now, by my reckoning, twenty something. Looking back, we see the 1980s as a period when considerable effort was devoted broadly to establishing conceptual foundations for the field. Researchers struggled with basic questions regarding the nature of work and how to make the user interface pleasant to use or "nonintrusive." In short, it was a time for constructing the CSE conceptual toolkit. Emphasis appeared to shift around the beginning of the decade of the nineties. With tools in hand, albeit sometimes in rough form, and with some design engineering experience, the deployment of CSE grew to a more sizable level of participation in large-scale projects and the development of support system prototypes to address hard and complex problem-solving situations in the context of ill-defined, high risk work. Thus, during this period we can see how the conceptual underpinnings of CSE are used to improve

design engineering tools and practices while simultaneously creating design innovations. In some instances, the coupling of engineering tools to theoretical development is tight and obvious; in others it is looser and less obvious.

It should come as no surprise that each of the four mentioned variants of CSE held different views on the conceptual and engineering tools needed in the field. The founding scientist-practitioners provided a hint of these differences by way of the variations in motivation for addressing cognitive work. These variations highlight the fact that CSE comes in different colors or flavors, based on different theoretical views. To contribute to the fundamental advancement of the field, one must understand these differences. Second, because the theoretical divide has proven to be quite large in CSE, if these differences are not understood the larger acquisition and development community will be less able to make informed judgments about how to capitalize on what CSE can offer to systems design and development. Third, there are differences and inconsistencies in how some common terms that span the entire field of CSE are used. These differences stem, in part, from the meanings ascribed to the same or similar terms by the different variants. Thus, practitioners and users alike can avoid confusion by having a better understanding of these nuances. In sum, variants within the CSE field define different frameworks that provide the conceptual landscape for the field. An inspection of these frameworks, therefore, provides a window into the maturity of the field: What types of frameworks have been established? What are the key differences between them? To what degree have they been formalized?

2.3 COGNITIVE SYSTEMS ENGINEERING GENOTYPES

The remainder of this section provides further elaboration of the various threads that serve to demarcate the CSE conceptual landscape. I call the frameworks that have evolved from the four identified starting points CSE genotypes, or simply genotypes for short.[3] Given the differences in theoretical stance and perspective among the four strains, this seems like an appropriate use of the term. Each genotype is labeled by using the name of the institution that reflects the affiliation of the principal champion at the time a strain was established, with one exception. The Woods group was originally formed at the Westinghouse Research and Development Center; however, over time this group generally has been recognized as associated with The Ohio State University; hence I will use OSU to label this brand of CSE. Following this scheme, the other three genotypes are CMU, RISO, and UCSD (see Figure 2.2). They will be discussed in alphabetical order.

3. Kim Vicente has also used the term genotype as a means of distinguishing between perpectives of CSE espoused by Jens Rasmussen and David Woods. Here I have followed his lead and adopted the term for a more comprehensive analysis of the field.

CSE Genotypes		
Name	Origin	Champion
CMU	Carnegie Mellon University	Card[4]
OSU	The Ohio State University	Woods[5]
RISO	RISO National Laboratory (Denmark)	Rasmussen
UCSD	University of California at San Diego	Norman

Figure 2.2: Naming convention to identify different strains or genotypes of cognitive systems engineering.

2.3.1 CMU Genotype

The expressed aim of this genotype is to establish a unified framework that is able to reflect scientifically founded knowledge about human performance in a manner that is directly useful for "engineering style" calculations used in the design and development process for interactive systems. To achieve this goal, Card and his colleagues proposed to develop "problem-oriented" models of human performance. These models would provide design engineers a vehicle by which they could easily represent specific work tasks in terms of human information processing. Ideally, a model would provide a quantitative analysis of projected human performance for any particular task. Thus, such a model could be used to make quantitative predictions with regard to design factors, or for the comparison of alternatives. In addition, human-centered models could also be used to help formulate alternative task structures in a manner that would make explicit the demands placed on human physical and cognitive processing resources. If task structure and system design constraints are already formulated, then the model could be used to access the physical and cognitive cost of achieving the task with those conditions.

To achieve this goal, Card, Moran, and Newell developed two cognitively oriented modeling frameworks; one directly focused on the human and the other focused on the task expressed in human problem solving terms. The Model Human Processor (MHP) uses knowledge, derived largely from cognitive psychology and cognitive science, to specify elements of an information-processing architecture in a computable form (Card et al., 1983, 1986). For example, the model specifies elemental components of and derives processing times for human perceptual, cognitive, and motor activities. The architecture considers the ability and constraints of the human to mentally chunk, forget, store, and retrieve information and to reason with information.

While it has become traditional to refer to architectures of this nature as cognitive models, in fact, they include perceptual and motor processes also. The emphasis placed on cognition is due to the fact that earlier information-processing models used in design engineering tended not to make cognitive fac-

4. Card was affiliated with CMU at the time he was working on the Human Model Processor and GOMS modeling methods. He susbsequently moved to Xerox PARC, but the CSE work has been sustained at CMU by others.

5. Woods and colleagues were affiliated with the Westinghouse Research Center when their work on CSE began. However, it has been sustained largely by Woods and colleagues at OSU since that time.

tors explicit, although there are some exceptions.[6] Models based on information theory or control theory, for example, consider the human as an (signal) information processor but the characterization resulting from these theoretical frameworks did not address details of cognitive work (D'Azzo & Houpis, 1966; Shannon & Weaver, 1949). Thus, CSE constitutes a "new look" at an old problem in human factors engineering: how to model the human in terms useful for design engineering. The new look from this genotype emphasizes the explicit modeling of mental operations, as opposed to using a single global construct like information. One could argue that this new look dates at least to Broadbent's (1958) seminal work on attention, but a more thorough cognitive development did not occur until Neisser (1967) and Simon and Newell's work on production systems (Newell & Simon, 1972). Card, Moran, and Newell (1983) represents the first time someone attempted to transition these new gains in understanding human cognition into engineering tools for the use of designing better interactive systems.

An important aspect of the MHP framework is that it provides a standard set of parameterized information processing primitives and a fixed process architecture that includes cycle times for three independent processes (perception, cognitive, and motor). As a result, it has the means to produce models of human behavior founded on established scientific understanding of human information processes. Card et al. (1983) carefully selected primitives that span information processing for an intact agent or user and established effective approximations and parameters for these components in a manner that was faithful to the scientific database, while simultaneously avoiding unsolved and controversial scientific issues in the study of human cognition. Because the model served to integrate, in process form, knowledge from three diverse areas of behavioral research, the MHP can be used to simulate complete human behavior. In effect, the model becomes a unified theory of behavior that emphasizes cognition. Thus, it provides the design engineer with a tool that can be used to analyze task work in terms of perceptual, motor, and cognitive factors. In this way cognitive factors can be included in the assessment of a range of system design issues.

The second modeling framework developed by this group goes by the acronym GOMS, which stands for Goals, Operators, Methods, and Selection rules. Whereas the MHP framework represents a synthesis of knowledge on human information processing, GOMS represents a synthesis of what is meant by the term work or task from a cognitive perspective. In general terms, work is conceptualized as being a goal-driven activity that is attained through rational action. Further, work is regarded as problem solving. These views are captured in the *Problem Space Principle* and the *Rationality Principle*. The Problem Space Principle is predicated on the notion that a person's activity can be characterized as applying a sequence of actions to transform an initial state into a final goal state. In other words, work is a problem to be solved, and problem

6. It is traditional to call models like the MHP cognitive models or cognitive architectures because of the emphasis placed on human thinking processes. They usually devote some attention to other aspects of information processing such as perception and motor processes as well, but the treatment may be less well developed. Thus, they may more correctly be called models of human behavior; however, we will follow established tradition and use the term cognitive modeling here.

solving is regarded as a search process that transforms an abstract, cognitive goal into a physical goal that can be obtained by a sequence of mental and physical actions. The second guiding concept, the Rationality Principle, asserts that a person will develop methods to become efficient at problem solving, given the structure of the task environment, human knowledge, and processing capabilities and limitations. The GOMS constructs implement these principles.

In GOMS, a goal is defined to be what a person is trying to accomplish at a functional level of description. Thus, it is a symbolic or abstract expression stated in natural language that specifies what to accomplish but not how to accomplish it. A goal statement, therefore, acts as a top-level task definition. (At this level of abstraction, a goal statement of work is equivalent to a functional statement of a system in user-centered terms.) The remainder of the GOMS model expresses how cognitive operators reduce this abstract understanding to actions taken in the world. The reduction process involves a cycle of cognitive activity that is conceptualized in terms of the goals, methods, operators, and selection rules. An operator transforms a goal into a subgoal that is "closer" to the desired end state. A sequence of subgoals is often needed to traverse the problem space. A method is a sequence of operators and subgoals to accomplish a higher-level goal. A method acts like a chunk of knowledge that can be activated as a single unit. Its formation and use is dependent upon prior learning. It may be thought of as a procedure. Selection rules represent knowledge about which method to apply when there is more than one method available to complete the task. They are the control mechanism for the modeling framework. The cycle of decomposition terminates when execution of a subgoal directly yields an elemental action that achieves the desired goal state.

To illustrate the GOMS framework, Card et al. used the problem domain of text editing. Consider the situation where one wishes to delete a phrase. DELETE–PHRASE, then, may be an expression of the task and, hence, the active goal of the agent. To achieve this goal, the agent invokes a series of operators, like: MOVE–MOUSE (to beginning of the phrase), CLICK–MOUSE–BUTTON, WIPE–LETTERS, RELEASE–MOUSE–BUTTON, HIT–DELETE–KEY.[7] With experience, one may chunk this sequence of operators into a method that can be called the *MARK–AND–DELETE* method. The text editor may support other methods, as well. For example, the same task could be accomplished by following a *DELETE–CHARACTER* method (i.e., deleting one character at a time). This method would have to be invoked n times until no more characters were present. Now, a writer (agent) may (implicitly) formulate a heuristic that says: use the *DELETE–CHARACTER* method if the number of letters is $\leq x$; otherwise use the *DELETE–PHRASE* method. This heuristic acts as a selection rule. A GOMS model for a task consists of a description of a sequence of operators, methods, and selection rules like this to achieve the initiating task goal.

7. For any specific model, the analyst describes operators at a level appropriate to its planned use.

Within this basic structure, a GOMS model incorporates information processing capabilities and limitations of the actor associated with activating goals, operators, methods, and selection rules. In its simplest form, an analysis of the cognitive demand of any identified GOMS model can be expressed in terms of cognitive processing time for each GOMS construct. In other words, each construct is treated as an activity that consumes human processing resources (cognitive, perceptual, or motor) that is expressible in units of time. Task time then depends on the set of mental activities included in the problem solving process. Prior to learning a method, the model predicts some task performance time *x*. After learning, one or more methods may be formed; the model predicts a new task time *y*.

The initial family of models developed by Card et al. was strictly hierarchal in form (i.e., a linear sequence of GOMS operators). Subsequently, the GOMS family has been expanded to include models that support parallel processes and a more detail explication of cognitive processing. These include CPM–GOMS and NGOMSL. CPM–GOMS represents parallel processes for Cognitive, Perceptual, and Motor (CPM) activity (John Kieras, 1996a, John and Kieras, 1996b). For a CPM–GOMS model, operators for each type of process are arrayed in time, based on meeting constraints of the task and the internal dependencies between operations within the agent. Further, all operators have a time parameter—how long it takes them to execute. The task is then analyzed in terms of interconnections across these three streams of mental or human information processing operations. An important aspect of the analysis is to inspect the set of operators to discover which operations of the different class types can occur in parallel and thus co-occur during a time interval. As a result of this arrangement, a specific type (cognitive, perceptual, or motor) of operator may be initiated and completed before another co-occurring operator of a different type completes its work. In effect, one operator may be completely or partially "covered" in time by another one. Therefore, total task time depends on an analysis of this complex web of mental activities. For example, an agent can be reading and processing text while simultaneously moving the mouse into position. The portion of mouse movement that co-occurs with text reading is subsumed by the longer reading time, and thus it is not reflected in the final time estimation for the task. In this way, a time- and constraint-sensitive critical path is formed in a CPM–GOMS model to predict task performance time. The acronym CPM carries the dual meaning of cognitive, perceptual, and motor, and critical path method in CPM–GOMS.

NGOMSL is a structured natural language notation for representing GOMS models that includes the constructs and form of a cognitive architecture called the Cognitive Complexity Theory (CCT) (Bovair, Kieras, & Polson, 1990; Kieras, 1988; Kieras & Polson, 1985;). The CCT includes mechanisms for internal cognitive operators for such things as adding and removing information from working memory or setting up subgoals. The notation provides a

structural way to organize external keystroke-level operators and the more detailed internal cognitive operators. The relation between the external task and internal human processing operators is direct and thus can be used to produce an integrated and computable production system (Newell & Simon, 1972). Hence, the NGOMSL system provides a means for generating GOMS models in computational form. The system can be used to represent a task at multiple levels of detail. Because of the requirements of the computational architecture, NGOMSL enforces a more rigorous development process for creating GOMS models than is true of the other members of the GOMS family.

What have we learned about the conceptual orientation of this genotype of CSE? Clearly the heart of this genotype is a modeling framework aimed at representing, in some form, the mental activities predicted to be involved in completing a task. In broad terms, the conceptual focus is on work as a cognitive activity. More specifically, the conceptual logic of the CMU genotype can be summarized in the following way:

- Agents engage in work
- Problem solving involves cognition (often a large component of work)
- Any work can be modeled as problem-solving activity
- Once a goal is established, problem solving can be achieved by a goal decomposition process (i.e., goal-bound task is the focus of analysis)
- Formal models of work can be formed within a GOMS framework
- These models can be analyzed in terms of cognitive activity, and
- Thus they can predict cognitive work based on a task design.

The formal constructs of the CMU genotype are contained in the GOMS formalisms. MHP concepts are loosely associated with the original GOMS formalism, but they are more tightly associated with the CPM–GOMS formalism. The analyst who builds a GOMS model is free to select the level of resolution at which an operator or method is expressed. In this way, a GOMS model can be expressed at a level of approximation that is deemed by the analyst to be useful to meet the needs of specific design engineering conditions. When appropriate, more detailed models can be generated.

It is useful to note that the modeling strategy treats elements of the *work domain* (features in the environment) as properties of *work activities*. For example, a MOVE–MOUSE operator incorporates a domain object in activity language. The mouse object is not considered independently from the mouse-moving activity. Thus, work domain items are included only as elements of activities in the formalisms used by this genotype. Other CSE genotypes follow strategies that place emphasis on preserving the distinction between work domain objects and work activity objects.

It is also important to note that this genotype implicitly regards work as a collection or set of tasks ($n \geq 1$). As stated earlier, a task is defined by the acti-

vation of a driving goal state. A task exists for the duration of time between the initial activation of a driving goal and its termination by achieving a final goal-satisfying state.

2.3.2 OSU Genotype

The impetus behind this genotype was the desire to explore the use of emerging computer technology to produce decision support systems to aid control room operators in the electrical power generation industry. Woods (1986) noted that expert systems, a popular application of artificial intelligence research in 1980s, considered decision support in terms of using computer (reasoning) automation to produce answers for user consumption. That is, a knowledge-based or cognitive machine was designed as a tool for use by a worker. The expert system takes input from the worker, uses its knowledge and produces an answer for the user, often after periodically asking the user to clarify or provide more information. The knowledge-based reasoning output generated by the expert system is then available for use by the human. Woods took the stance that cognitive tools would receive better reception and provide more effective aiding if they worked as partners *with* the human worker in the process of decision-making and problem solving. In other words the human and the machine both develop an understanding of the problem and ways to solve it as a dynamic process, as opposed to the human feeding the expert system and digesting its response. This stance represents a shift from viewing aiding as an outcome that is provided by the intelligent machine toward a view of *aiding as a dynamic work process* involving both sources of cognitive ability—humans and smart machines. Because the process is dynamic and subject to unexpected events, this shift implies that a different kind of relationship must be established between the human and the decision-supporting tool. Hollnagel and Woods (1983) used the term *joint cognitive system* to distinguish this view from the outcome-oriented expert system approach.

Woods noted that workers utilized properties of the workplace, including engineering tools, as cognitive aids. This implies that cognition extends to artifacts in the world, as well as the mental abilities of the worker(s). Therefore, in some sense, cognition is distributed: it is both in the head and in the tools. Thus, to model cognitive work, one must model aspects of the work world and the worker in cognitively relevant terms. Further, this also implies that cognition is situated. Change aspects of the work world and "cognition" (information and its meaning) may also change. These observations, and others, drove Woods to analyze work *in context*. In other words, the interaction between the worker, the work environment and tools in the environment represent the appropriate unit of analysis. This has implication for how one represents the fundamental nature of work, and thus, how it can be supported through design engineering.

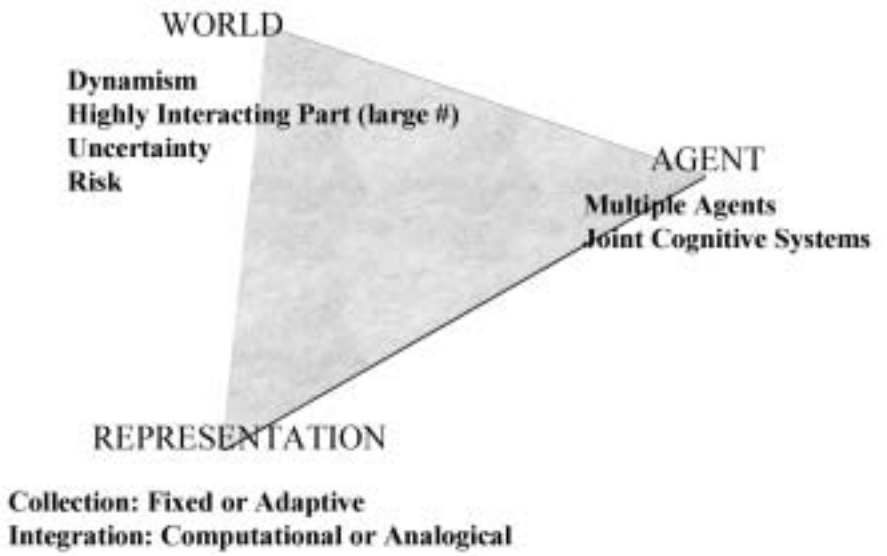

Figure 2.3: The cognitive triad model of work-in-context. (Adapted from Woods & Roth, 1988).

In an effort to represent work in the cognition-in-context framework, Woods developed the cognitive triad concept. This looks at work in terms of the *demand characteristics* of the work world, and considers work as problem solving. As shown in Figure 2.3, the triad is composed of agents, representations, and the work world. The triad represents the problem-solving space in terms of the interactions among agents, representations made visible in the human-tool interfaces, and the work world. The cognitive demand of the world is represented in terms of its dynamism (i.e., rates of change), the number of parts of a problem, and the extensiveness of interactions among parts, uncertainty, and risk in the situation. Risk expresses cognitive demand of the world in agent terms, while uncertainty may be due to both agent independent or agent relative factors. Taken together, these dimensions of the work world serve to characterize the complexity of the world that at once becomes part of the problem to be solved (i.e., the distributed work goal) while acting as impediments to goal satisfaction (i.e., constraints on the driving goal as a problem).

The representation construct addresses the fact that how a problem is expressed in a medium makes certain information or manipulation explicit and available to the agent at the expense of other information available in the world. Therefore, a representation invites a way to think and proceed, even though it does not preclude other ways to conceptualize and act. Thus, it makes some things apparent and cognitively easy and others opaque and cog-

nitively hard. There is ample empirical data to support these aspects of work (e.g., Norman, 1993; Wason & Johnson-Laird, 1972; Zhang, 1992).

The agent construct of the triad acknowledges internal cognitive factors (e.g., memory, reasoning skills, etc.) that impact how an agent interacts with tools and other properties of the work world. These are internal cognitive resources and constraints that must be considered in problem solving.

The cognitive triad represents an embodiment of the concepts, beliefs, and assertions of the OSU genotype. From this orientation, the field of study of cognitive systems engineering, in the broadest possible terms, is *humans at work with tools*. It asserts that worker behavior is influenced by the need to cope with the demands and constraints of the problem-solving world. Thus, work is taken to be problem solving *with tools*. Second, coping with the complexity of the work world itself is part of problem solving. Even though these aspects of the work world are incidental to the purpose-driven problem that sets the overarching goal that guides work in the first place, they are nevertheless part of the problem confronting the worker. This implies that the demand characteristics of the work place must be included in any model of work. The cognitive triad provides a theoretical framework that is able to capture these aspects of work in a general, context-independent way. Hence, it supports a scientific study of work conceived in this manner.

The representation and agent legs of the triad may also be viewed as expressing demand characteristics. By following the same logic, it is reasonable to assert that demand in both of these domains comes from *intended* and *unintended* sources, in the same manner as that found in the work world. Thus, like the work world, the representation(s) of the world available to the worker and the worker's internal demands both contribute to the solution of the work problem and constitute part of the work problem.

Figure 2.4 illustrates this point. The driving goal establishes the external demand from the world, as previously stated. The window to the world provided by tools, such as a computer or remote sensor, intends to make explicit information germane to solving the driving problem or goal. In this sense they are "demands" on an agent to use the available information for this purpose. Further, the agent is motivated and poised to deploy its resources, cognitive and otherwise, to use the available information in the service of solving the driving problem. Hence, these may also be regarded as positive "demands" on the worker. Now, as shown on the other side of the chart, these same sets of sources unintentionally introduce incidental factors that serve to generate negative "demands" that become part of the work problem. The world, for example, may include an array of items that, say, impede movement to a goal location. In a similar way, a representation of the world may be insensitive to certain changes and objects and thus add complexity for the human by requiring reliance on memory to keep them present because they are not immediately available for perception in the representation. The representation, in effect,

	Demand Characteristic	
	Incidental	Intended
World	Context factors	Problem goal
Representation	Hidden data	Direct information
Agent	Collateral states (e.g., emotional control, fatigue, etc.)	Goal-driven motivation; resource focusing

Figure 2.4: Multiple sources of cognitive demand. Some stem directly from the work problem and thus are "intended." Others result froom incidental factors. The locus of the source may be in the world, the representation used in the support artifact, or from the agent.

becomes part of the problem by what it makes opaque. In the same vein, the agent brings demands that unintentionally interfere with main-line problem solving. These include things like personal goals, adverse emotional states (and the ability to control them), and adverse physical states (e.g., fatigue) that all serve to "conflict" with the driving goal of the work problem. These unintended demands act to add complexity that the agent must work around by managing them. Hence, they are part of work even though they are not part of the initiating, driving problem. Thus, the world, representation, and agent are resources to solve a problem, and at the same time, they may, and generally do, add complexity to its solution. Both of these aspects are therefore included in the definition of work.

Pushing the demand language this far may seem a bit forced. Indeed, Woods does not make the incidental and intended demand characteristics distinction explicit, as done here. However, it serves as a useful way to make the point that the complexity of work comes from both factors directly associated with the driving goal of work, as well as from incidental factors that are distributed across the agent, representation, and other elements of the work world. It is for this reason that interactions among these constructs are treated as the unit of analysis by this genotype. In essence, there is a distributed network of goals involved in problem solving work. Thus, to model work, one must account for these nonproblem driven aspects of problem solving.

It is important to recognize that the OSU genotype views decision making and problem solving as related and open activities. That is, in a real-world context, the problem or decision requirement is subject to change while one is working on a solution to the driving problem. Other events occur that may change an aspect of the problem or the very nature of the problem itself. It is in this sense that problem solving is open: The "problem" is not a well-defined stationary task to be completed. It is in part because problem solving is open that we must understand cognition in context, or situated cognition, to design effective support systems to aid users in cognitive work. This view of problem-

solving work clearly contrasts with the view of work from the CMU genotype. A task-based model treats work as closed by limiting its definition to the driving goal. Thus work, as a technical construct, means different things to different cognitive engineers.

In addition to the cognitive triad, the OSU genotype has introduced a set of three interacting and partially overlapping constructs to represent factors that govern a human's intention to act in a work situation. These factors are (1) available knowledge, (2) attentional dynamics, and (3) strategic factors. These constructs were derived from an analysis of the origin of errors and the recovery from errors that have been observed over different work domains, ranging from commercial aviation to the medical operating room. Each factor consists of a set of cognitive variables that can be used as the basis for assessing properties of the work environment, including support machines and tools, and social systems that make work hard or error prone. For example, six cognitive variables have been identified under the knowledge factor: buggy knowledge, mental models, knowledge calibration, inert knowledge, simplifications and heuristics, and imprecise knowledge (Cook et al., 1992). Attentional dynamics deals with traps that invite inappropriate fixations and attentional shifts associated with establishing and maintaining situation awareness. Strategic variables include goal trade-offs and decision choices, including risk assessment. All three types of factors are implicated in the dynamics that shape work performance and that can be used to account for how errors arise in work. For example, Cook and colleagues illustrate how the design of a procedure for setting up an automatic infusion pump controller adds to work complexity and invites an error to emerge due to imprecise knowledge, memory lapse, and poor situation variables (Cook et al., 1992; Woods et al., 1994).

In sum, the theoretical emphasis of the OSU group has been focused on answering the question: What is the anatomy of the cognitive systems engineering problem? They have concluded that the scientific problem is the study of work as problem solving with tools. The engineering problem is how to design tools as (decision) support systems to aid the problem-solving work process. Their analysis indicates that work is open, cognition is distributed, and that dynamical and uncertain factors influence work. Rather than work being defined by the desire to achieve a single, task-defining goal, work is defined by a constellation of interacting goals incident to the work world. The cognitive triad is offered as a general framework to express the work problem to be solved. The knowledge-attentional-dynamics-strategies scheme serves as a general framework that identifies factors that influence intention formation during the course of work.

It should be obvious that these conceptual framing tools of the OSU genotype do not offer a well-formed specification for a process architecture of cognition. Rather, they serve as a context independent language that can be used to represent the work problem in an open manner. In effect, the OSU approach

regards the *actual* work problem to be emergent, and therefore, it is not appropriate to provide a specification for a process architecture to directly express the work problem. Accordingly, they offer frameworks that capture the problem indirectly. This is in marked contrast with the CMU genotype. GOMS is a specification for a process architecture for how to solve a work problem. The OSU genotype would claim that GOMS is a specification for how the *driving problem* can be solved, not a specification for the actual, emergent problem to be solved.

Finally, the OSU genotype may be regarded as an *ecological view of work.* Work is a dynamic process that involves the coevolution of factors from the world, tools, and agents. Again, it is useful to recognize the consequence of this technical distinction for how work is defined. A task approach begins with an analysis of work in the world, as would the ecological approach. However, the goal of the analysis for the task approach is to develop a model of a task that can be adequately expressed in terms of the driving goals or functions to be achieved. In this way the work formulated as a task can, in effect, be "lifted out" of the environment for use in a cognitive model as a basis for analysis. This, in turn, is used to support decision making about artifact design. The OSU genotype has a fundamental disagreement with this position. A single goal-driven task model of work does not preserve the dynamics of the work world. To preserve the open, dynamical attributes of work, it must be modeled in a way that maintains connection of the driving task with the constellation of goals embedded in the dynamics of the work world. A different type of "cognitive language" is needed to properly model work from this ecological perspective. As a result, it would produce a different kind of process architecture for a computational model of work.

2.3.3 RISO Genotype

The RISO genotype has much in common with the OSU strain. It too considers cognitive systems engineering from an ecological perspective. Work behavior is regarded as being shaped by factors in the environment, including the tools available to support work. Problem solving and decision-making are regarded as open processes, and cognition is distributed. Because work is open, in that it must contend with unplanned and unexpected conditions, a worker must be adaptable. This genotype focuses on adaptive behavior and how to support it. Its goal is twofold: to provide an engineering design framework that is able to effectively guide the development of an interactive system that is, first, robust to disturbances and second, error tolerant.

Support for adaptive work behavior is viewed as key to meeting these engineering goals. A system can be made to be robust to disturbances or unexpected conditions if it can behave adaptively in a way that preserves system stability while the driving problem is being prosecuted. Similarly a system can be

robust to error if it can make adaptive adjustments both to detect anomalies and then recover from them during the course of working toward the solution of the driving problem.

Like the CMU and OSU genotypes, the RISO genotype mainly considers work from a decision making/problem solving perspective. That is, the worker is seen as an adaptive decision maker or problem solver. Conceptually, work is considered to be both a component of a sociotechnical system and a characteristic of an agent. Work is both a part of the environment or work domain and the activity of an agent. In other words, the driving work goal is embraced by an agent and comes from the environment. The actual work reflects emergent behavior of the agent. The driving goal and other properties of the environment act as work goals or constraints on work that shape, but do not dictate, actual work. It follows from this view of work that any framework that can be used to model work must represent both the work domain and work activity, and the model must be sensitive to the dynamics in both aspects to understand what makes human work behavior robust and error tolerant.

Rasmussen has proposed a comprehensive framework that treats the work domain, work activity, and the work agent in separate subframeworks that are related to each other. The relation between frameworks acts like a process architecture for a nonlinear dynamic modeling system of work. As a result, the overall framework provides a detailed specification for modeling work, even though it only models work indirectly. This will be made clear in the following discussion.

The conceptual toolkit for the RISO genotype is organized around five interrelated frameworks developed by Rasmussen and his colleagues. For the purpose of discussion, I will refer to them in the following manner:

- Work domain
- Control task (CT)
- Control strategy (CS)
- Social/organizational (SO), and
- Worker competencies (WC).

2.3.3.1 Work Domain. The work domain framework provides a specification for how to model the field of practice in a way useful for design engineering. It proposes to model the work world in terms of two dimensions: a structural abstraction hierarchy and a whole-part or aggregation dimension. Together, these dimensions form a lattice that identifies the aspects of the work domain that are hypothesized to be important to guide the design of interactive systems.

The Abstraction Hierarchy (AH) identifies different types of functional properties of the work domain. A prototypical AH consists of five levels that follow a defined order. Somewhat different generic labels have been used to identify these levels over the years (Rasmussen, 1986, Rasmussen et al., 1994;

Functional Purpose	Domain Properties Represented
Purposes and Constraints	The purposes for which the system is being designed
Abstract Functions and Priority Measures	The intended casual structure of the work environment represented in terms of the flow of values and abstract physical properties
General Functions	Description of the basic process of the system in functional language
Physical Processes and Activities	Characteristics of the physical components associated with these processes and their connections
Physical Form and Configuration	Characteristics of appearances and spatial distribution of physical components

Figure 2.5: The abstraction hierarchy.
(Based on Rasmussen, 1986; Rasmussen, et al., 1994; and Vicente, 1999).

Vicente, 1999). The most recent generic labels are shown in Figure 2.5. The purposes and constraints level of the hierarchy is the place to represent what I have called the driving problem or goal of the domain[8]. It specifies the functional purpose(s) of the work domain. Goals are stated in terms of desired system outcomes. The priority measures level is the place to represent the values of the specific work domain in question. Values act as requirements derived from many different sources. They identify what the work environment regards as important; hence they are given priority. Typical sources of values include management/organizational; social, influence of government and regulatory bodies; and the constraints that derive from natural laws relative to product production (e.g., physical laws of nature). The strength of these value-based requirements generally will be different, and the perception of importance by the problem solver may vary throughout the course of completing an epoch of work. The general function level is used to represent the domain in terms of the general functions that must be achieved to yield the outcome product. It expresses a typical functional model for a system that identifies a set of necessary and sufficient factors to achieve the driving goal, without specifying the form of work production. The physical processes level is used to re-express the more abstract general functional model in terms of the physical variables available in the environment that can be used to implement the general functions. The last level, physical form, depicts the physical spatio-temporal layout of the workspace and its contained resources. It makes explicit the means by which the items at the physical function level of the hierarchy can be achieved. Thus, the physical form model may be regarded as providing objects expected to be

8. In the RISO genotype, the driving problem is considered to be a component of the work domain. At the highest level of abstraction, this problem is expected to map one-to-one and onto the driving problem adopted by a problem solver. In other words, the driving problem acts like an abstract task goal for an agent, as in the CMU genotype, but it remains as a property of the domain. Thus, there is a subtle distinction in the meaning of a driving problem when it is considered from the perspective of the RISO or CMU genotypes.

implicated in work (i.e., the work space as "figure") against the physical layout of the work site (i.e., the work space as "ground").

The second dimension of the work domain framework treats the domain as a system expressed at different levels of detail. It encodes a whole-part relation and is called the aggregation dimension, or as some prefer, the decomposition dimension (Eggleston, 1998; Vicente, 1999). When combined with the AH, the full lattice contains a family of languages, each of which concentrates on the domain as a system in terms of a different aspect, and each aspect is expressed at multiple levels of detail. In other words, the terms used in each row of the abstraction decomposition space serve to form a lexicon of a simple, single-focus "language." The constructs of the language compose a consistent set, with respect to the referent used to define each type of row. Because of this consistency, these sets are well formed and thus if used can help avoid logical paradox, such as the classic barber paradox. The total set of languages helps to provide a more complete picture of a work domain while preserving the logical distinctions among the languages.

A full Abstraction Decomposition Space (ADS) is generally quite large. For the purpose of illustration, only a highly compiled version of an enroute air traffic control domain is shown in Figure 2.6. More detail ADS descriptions would be included in a fully developed model (for more complete examples, see Bisantz & Vicente, 1994; Vicente, 1999).

Our interest for this discussion is on the structure of the framework and its use, and not on the details of the content for the framework for a specific work

	Abstraction-Decomposition Space		
	Decomposition-Dimension		
Abstraction Dimension	System	Subsystem	Component
Purpose	Aircraft Flow Control		
Priority Measures		Speed/Safety/ Economic Topology	
General Function		Aircraft Passage Flow Through Sector	In-bound; out-bound; and internal passage flows
Physical Function			Individual aircraft control states
Physical Form			Physical arrangement of transiting aircraft

Figure 2.6: A notional abstraction decomposition space for an enroute air traffic control work domain. The ADS is formed from a decomposition dimension and an abstraction dimension that follows the abstraction hierarchy.

domain; therefore, the discussion will be limited to these aspects. There are a few things to note. First, it is customary to leave some cells open. For example, there is generally little or no value to providing a highly compiled system level representation of the physical layout. Smaller scale objects are needed to show the important aspects of the spatial and temporal configuration of the workspace. The same holds true for the physical functions. It is important to individually represent the physical variables that can satisfy different functional uses. Thus, they are naturally expressed at a subsystem or smaller-size scale. Only cells useful for design decision-making are populated.

Second, there are many situations, especially early in the work analysis process, when the physical form level can be omitted from the representation without causing a serious loss of useful information. Third, the top four levels of the AH should be taken as the minimum number of levels needed to construct a "full" AH. In other words, the aspects of the work domain reflected in these levels are taken to be necessary to produce an adequate model of a field of practice. Some researchers appear to believe that the necessary and sufficient set of levels to represent a domain varies by domain. Others suggest that the five levels constitute both a necessary and sufficient set. How many levels are required and what criteria need to be used to select the appropriate number for a specific domain remains an open issue. No one has provided a logical or empirical basis that supports the five-level or open-size positions.

A fourth important point about the work domain framework is that it is an object-oriented representation. Even though it describes functional properties of the field of practice, the expression is in a structural form. It does not specify a process model. Rather, it specifies a structural model of the functional landscape relevant to interactive system design. As a result, every element is an expression of work in an event-independent form. It represents a complex structure or domain where activity takes place. It is both where work takes place and a source of resources and constraints that serves to provide the complex texture that emerges in work activity.

Finally, it is important to recognize that each abstraction level serves to completely represent the entire system or domain, in terms of the aspect being expressed. That is, each level or row of the ADS provides a separate and complete model of the work domain. Thus, the entire framework includes a minimum of five interrelated and complete descriptions of the work domain.

2.3.3.2 Control Task. The control task framework of the RISO genotype considers the use of the work domain in activity language. The functional purpose of the domain establishes the necessity for a control task to be accomplished. The control task framework is used to characterize the structure of the event-dependent process to achieve the domain goal. In other words, the control task language is used to describe what needs to be done to achieve the goals of the

work domain, within the prevailing resource and constraint characteristics of the domain. Prototypical processes are generated and expressed as categories of possible ways to accomplish work, as defined by the driving purpose.

Rasmussen has developed a decision ladder to act as the framing tool for control task description (Rasmussen, 1986; Vicente, 1999). As shown in Figure 2.7, the decision ladder characterizes a control task in terms of information processing or cognitive activities and knowledge states. It is important to note that this is not representing the information processing of any identified agent. It is a representation of goal-driven work in activity terms, regardless of who (human or machine) accomplishes it. In essence, the decision ladder takes linear information processing stage model and bends or folds it around the evaluate performance criteria activity, thereby creating an inverted "V" that resembles a ladder. The left side of the ladder considers work in terms of sensory and perceptual activities. The right side considers work in terms of problem-forming and execution activities. Again, it is important to keep in mind that even though the information processing language being used in the decision ladder is normally associated with human information processing, the framework focuses on the *nature of the control task(s)* of the work domain. It is not intended to be a model of how a human or any other specific agent accomplishes the task.

Like the work domain lattice, the decision ladder is a template that can be populated by an analysis of a specific control task. For a specific task, labels for the process activities and knowledge states are used that are relevant for the context. In addition, short cut paths or shunts that are supportable by domain resources, including resources for specified classes of agents, are also expressed in the decision ladder. Labeled arcs between data processing/knowledge state nodes are used to depict these aspects of supportable information processing behavior. For example, an agent may observe data from the environment and based on internal resources (e.g., memory) may leap to the conclusion that a specific task requires action and then proceed to formulate a procedure to accomplish the task. Once populated with context-specific information and known or asserted capabilities of different classes of information processing agents, an activity-based model of the nature of work can be mapped onto the decision ladder template.

As the last paragraph implies, the nature of a control task is not permanently fixed for a work domain. Rather, it is relative to the resources available in the work domain and thus, changes as these resources change. This, of course, includes the resources provided to the domain by any class of agent. It follows that the nature of a control task is more difficult to describe adequately as more resources in the work domain are able to dynamically change themselves. In principle, the decision ladder provides a modeling tool that can address these dynamical properties of work activity in a systematic but open manner.

According to the RISO genotype, an *actual control task form will be emergent.* However, it is expected to have the "stable" characteristics derived from

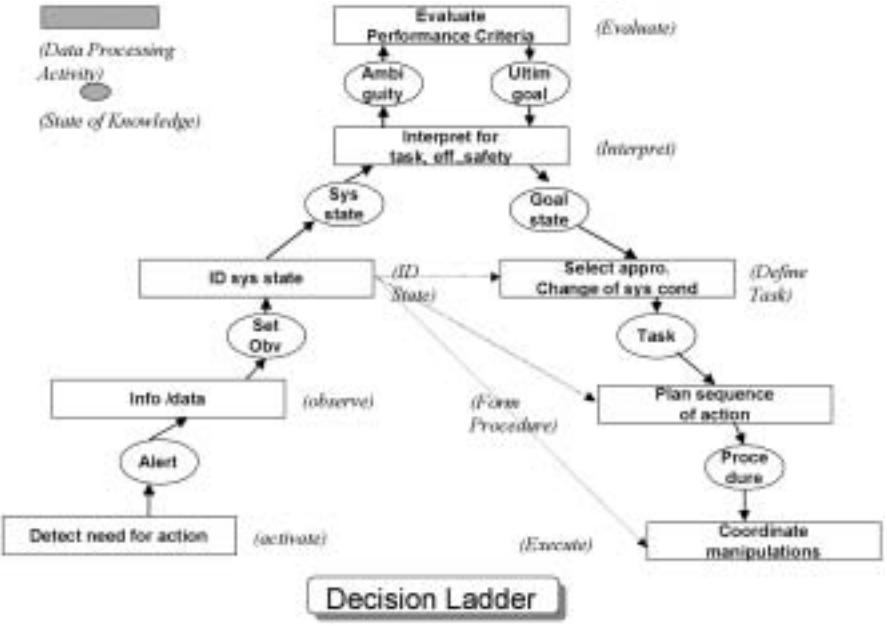

Figure 2.7: The decision ladder modeling tool used to represent work activity in information processing terms. Prototypical, expected, or observed information processing short cuts can be depicted by different style arrows. Three shortcuts are shown for illustration. (Adapted from Rasmussen, et al., 1994).

the driving goal of the domain, relative to possible agent abilities and constraints of the domain.

2.3.3.3 Control Strategies. The control strategies framework follows in the same spirit as that of the control task. It seeks to represent, in an open manner, the intrinsic nature of work strategies that can be used to accomplish a control task. The difference between the decision ladder and the control strategies frameworks is simply one of focus. While the control task model provides the basic shape or shapes of problem-solving work, a control strategy model provides different classes of methods by which these activities may be addressed by a class of agent. Each strategy class requires certain information-processing resources to accomplish the control task in a specified manner. Thus, in this way strategies serve to specify information requirements for agents that can accomplish the control task in different ways.

Rasmussen takes the position that a control strategies framework should be used to represent different reasonable ways an expert could work on the work problem. In other words, even though the control task and control strate-

gies frameworks are open and thus allow an infinite number of strategy types to be formed, for the purposes of guiding design engineering decision making, Rasmussen et al. (1994) recommend analysis be concentrated on the discovery of strategies employed by experts in the work domain.

A modeling tool used to express control strategies is the information flow map (Rasmussen et al., 1994). Vicente provides a nice illustration of how a form of a flow map can be used in this capacity (Vicente, 1999). Strategies may also be expressed as information processing modes and thus can also be represented by a decision ladder.

2.3.3.4 Social / Organizational. The social/organizational framework is like the others in that its purpose is to reveal the structure of the work domain in terms of its social and organizational properties. This includes how work is partitioned among agents and the type of leadership style that pervades relationships and interactions throughout the organization. These factors influence both the necessity for communication and the structural form communication may take, as well as constraints derived from different forms of communication. Both social and organizational factors add to the demand characteristics of the environment, and thus become part of the intended and unintended aspects of the domain that contribute to the nature of problem solving work (Rasmussen et al., 1994; Vicente, 1999).

2.3.3.5 Worker Competences. The last conceptual framework in the RISO genotype is known as worker competencies. As suggested, this framework is used to represent the capabilities and limitations of different classes of agents, human or machine. Again, the objective is to capture intrinsic agent constraints relative to the various shaping factors from the previously developed frameworks. In other words, the other frameworks provide insight into the form of constraints that have implications for what kinds of competencies of an execution agent should be needed to successfully adapt to contingencies and achieve the driving goal. Toward this end, Rasmussen (1983) devised the Skills-Rules-Knowledge (SRK) taxonomy as a framing device. This taxonomy consolidates agent information processing into three alternative cognitive processing modes. Each term in the SRK label reflects one of these control modes: skill-based, rule-based, and knowledge-based control. The knowledge-based control mode is used to represent all forms of deliberate reasoning. Using stored knowledge to form an analogy or logical syllogism is an example of deliberate reasoning. The rule-based control mode, as the name implies, represents rule following. It assumes the existence of rules that can be recovered from memory or activated by recognition (e.g., if the light is red, then stop the car.). The skill-based mode represents control following "natural laws"

(Rasmussen, 1986). An example is the ability to make adjustments to your car's position by noticing a loss of safe distance with the automobile in front of you. The natural law is carried by the perceived rate of closure expressed in retinal size information naturally conveyed by the structure of the leading automobile. In general, the three modes represent different ways the worker interprets information from the environment. It gives recognition to the fact that an object in the environment may have multiple meanings based on how it is interpreted. The skill-based, rule-based, and knowledge-based modes reflect differences in agent processing required to achieve an interpretation based on the communication form ascribed to an object. Knowledge-based control asserts controlled symbolic processing. Rule-based control asserts perceptual sign or cue processing that activates stored rules. Symbolic and sign-based interactions with the world reflect an arbitrary mapping of objects in the world to the symbol/sign interpretation languages (Rasmussen, 1983). Skilled-based control asserts the use of energy-based signals that lawfully map domain objects and events to the agent. Thus, in Rasmussen's terms, when an agent is able to operate based on the pickup of affordances in the domain, it is working under skill-based control (Gibson, 1979).

The key components of the RISO framework are the work domain, control task, and worker competencies subframeworks. All of the frameworks are related. The work competencies maps onto the control task, and the control task maps onto the work domain. In this way, event-independent and event-dependent representations are brought together to produce a kind of "process architecture" for this indirect model of work. An emergent process is dynamically formed between the work domain properties and the work activity properties relative to an active (worker) agent. Like the OSU approach, when populated for a specific domain, the set of frameworks that comprise the RISO genotype express the demand characteristics of the field of practice. This representation goes beyond an expression of the work field as a driving goal to be achieved, as in the CMU approach. As a result, a larger set of factors that contribute to problem solving complexity are included. Thus, the representation is able to account for both intentional and unintentional, or expected and incidental, demands that serve to make problem solving hard.

As indicated at the outset, The RISO genotype is concerned about adaptive problem solving behavior. Because the work domain framework represents multiple aspects of the field of practice, it is able to serve as a comprehensive domain map. As a result, if perturbations or unanticipated events occur, the degrees of freedom in the domain are available as potential (cognitive) resources that can be used by the agent in work activity to form an appropriate adaptive response. While normally not stated, the same argument applies to the other representations in the total framework. The control tasks and control strategies models also include degrees of freedom that can be exploited to meet unexpected demands. In this way the RISO analysis system is able to address adaptive behavior that is

"error tolerant" without specifying a specific adaptive process to be initiated based on specific work conditions. It models work as a self-organizing system using resources in and being constrained by the work environment.

The total framework may be regarded as a nontraditional specification language for a process architecture of work behavior. It is nontraditional because it tries to specify a self-organizing system within a constraint space. As a result, the specification lays out the interacting shaping factors and does not directly specify the emerging work process. Rather, it attempts to specify what are believed to be the *intrinsic characteristics of work* (Rasmussen, 1994; Vicente, 1999) that are largely, if not completely, responsible for work behavior. An analyst who fills in the framework for a selected domain completes the framing language. This is similar to how the GOMS framing language (or architecture) is used, as is also the case for the less developed cognitive triad language of the OSU genotype.

2.3.4 UCSD Genotype

The fourth genotype considers the field of CSE as a new joint science-engineering discipline to guide the design, construction, and use of systems. Its expressed goal is to produce, in a principled way, the ability to effectively relate relevant physical variables of the system to the psychological variables of the human user of the system. To achieve this goal Norman asserted that the field first needs theoretical tools to "understand what the user is doing" (Norman, 1986, p. 37). Thus, like the other genotypes, it recognized the need for conceptual tools that focused on understanding human work. Norman proposed a seven-stage model of activity as a framework to represent work activity. Like the CMU genotype, this framework is expressed as a specification for a goal-directed activity process. The seven-stage model is depicted in Figure 2.8.

This model includes both psychological variables and physical variables. The goal is to be able to link the physical variables to the cognitive variables. The seven variables depicted above the horizontal line express mental or cognitive activity. Physical activity, shown below the line, directly relates to physical states of the system and expresses an eighth variable class of the framework. As presented by Norman (1986), the *goal* construct represents the state the person wishes to achieve. An *intention* is treated as a mental activity that constitutes a decision to act to achieve the goal. The *action specification* construct is regarded as "a psychological process of determining the psychological representation of the actions that are to be executed by the user on the mechanisms of the system" (Norman, 1986, p. 37). The *execution* variable produces a mapping from intentions to desired system state that serves as a mental specification of the actions to be physically executed. *Physical activity* relates to the physical state of the system. *Perception* refers to the translation of the physical

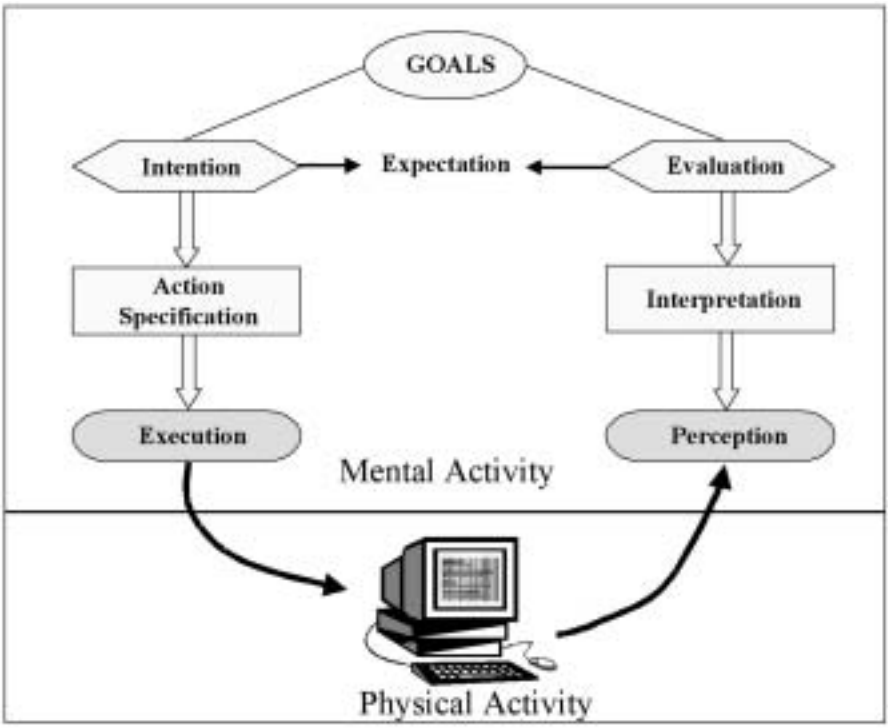

Figure 2.8: The seven-stage model of activity. (Adapted from Norman, 1986).

state of the system into the psychological state of perception that involves a layout of objects on a surrounding surface. It includes all perceptual operations. *Interpretation* is the process of converting the perceptual state into meaning with respect to the goal and intention(s). *Evaluation* is a process that compares interpreted meaning with active intention(s) and goals. The entire activity process iterates with new goals and intentions being formed as a natural outcome of the evaluation subprocess.

Like the GOMS modeling framework, the seven-stage model specification is expected to be used to produce an approximate model of problem-solving behavior. While it is presented in linear stage form, specific instances may include repeated and skipped steps. As a result, it is not as procedualized as the GOMS family of models.

In contrast to the other genotypes, the UCSD strain tends to focus on work in terms of direct interaction with a computer system. In other words, work is viewed in terms of human-computer interaction. The physical variables refer to the physical properties of the computer interface when work is accomplished directly with a computer. It is also clear that the specification for the seven-stage model is not as well developed as that for a GOMS model. The seven-stage

model provides cognitive constructs, but the actual process is sufficiently underspecified in that it does not constitute well-formed process architecture.

One use of the seven-stage model is to directly support interface design for interactive systems. The UCSD genotype has considered the cognitive nature of this design problem in terms of two gulfs, the gulf of execution and the gulf of evaluation, that separate the cognitive variables from the physical variables of the interactive system. The gulf of execution refers to the demand of the interactive system on the user to turn (cognitive) intentions into input to the computer. That is, the user, who thinks in terms of a "work language," must translate intentions into the "interface language" of the computer to use the computer in work. The gulf of evaluation is the demand on the user to interpret computer output relative to work need. The seven-stage model provides the designer with a device that can be used to make the cognitive nature of work explicit to better understand these demands.

To bridge these gulfs, the cognitive problem can be divided into two pieces. Each piece is associated with a specific pair of cognitive stages. These are known as the semantic distance and the articulatory distance problems. The semantic distance, on the input side, refers to the cognitive separation between the work-centered language of user intention and the semantics of the user interface language that must be used to enter the intention into the computer (e.g., menu labels, text input, etc.). On the output side it is the distance of the computer language to the cognitive stage of evaluation (e.g., actual content change versus desired change). Thus, semantic distance addresses the *content* problem of the user interface relative to the user's work focus, as reflected in the cognitive activities of intention and evaluation formation.

The articulatory distance addresses the *form* of expression of the information by the computer. A form mismatch on the input side increases the load on the action specification stage of cognition. On the output side, it increases the load on the interpretation stage. Thus, if the form of expression in the computer interface does not match the form of action specification or interpretation, then the user must overcome this distance by engaging in more cognitive work. It follows from this view that there are at least three cognitively inspired design goals:

- Minimize semantic distance from system operation to work
- Minimize articulatory distance from the system to work, and
- Maximize direct engagement with work.

The first two goals follow directly from the previous discussion. The third goal introduces another construct in the language of the UCSD genotype. Direct engagement is related to but different from the construct of distance. Engagement is said to be direct when the user's interaction with the computer results in a feeling of involvement with objects of the work domain (Hutchins,

Hollan, & Norman, 1986). In other words, as an engagement becomes more direct the computer effectively becomes more invisible. The user feels like he/she is doing goal-driven work rather than manipulating a tool to indirectly accomplish goal-driven work. Whereas semantic and articulatory distance are defined in relation to the operation of the computer as a tool in work, direct engagement is defined in relation to goal-driven work itself. Together, these constructs capture the demand characteristics of the problem-driving goal and those associated with the incidental demand characteristics (from the computer), in the same spirit as the OSU genotype. The major difference between these genotypes in this regard is essentially the scope of demand characteristics that can be explicitly considered. The UCSD genotype seems to limit incidental factors to the computer system, perhaps on the assumption that computers will usually mediate knowledge work.

The UCSD genotype is like the CMU strain in that it follows a cognitivist orientation toward work. The seven-stage model of activity is a process-based framework. It attempts to model work directly as an explicit process. However, in its current form it falls short of being a cognitive architecture because control mechanisms have not been adequately specified.

At the same time, the UCSD framework may also be regarded as considering the work domain in structural terms. The notions of semantic distance, articulatory distance, and direct engagement speak to the work domain as challenges to be addressed. Thus, like the RISO framework, separate models are used to capture aspects of work activity (seven-stage model) and the work domain. However, it is again clear that the conceptual formulation from this more ecological perspective is not developed to the same extent that it has been in the RISO genotype.

The UCSD genotype is closely associated with user-centered design of the human computer interface. This point is made clear by Norman's comments on the science of CSE. In his terms, CSE is about creating a science of user-centered design. This science produces principles that can be applied at the time of design and yields designs that are pretty good at meeting the three previously stated design goals during the first design iteration. Thus, CSE science includes a study of work, human activity, and how to convert this knowledge into useful design principles to impact design decision making. CSE is the science of work relative to design engineering.

2.3.5 Conceptual Distinctions

These four genotypes provide a wealth of conceptual distinctions that reflect the study of cognitive work in CSE for the express purpose of including better descriptions of cognitive factors in systems design. They make clear the difficulties involved in modeling work in a consistent, complete, and useful manner.

They point out that there are at least three alternative ways to consider cognitive work, all of which may be useful for engineering design. Cognitive work can be viewed as (1) a property of a worker or agent, (2) a property of a purpose-driven task, or (3) a property of a web of factors that defines the work domain, including the agent or agents. Although it is clear that there is no single way that can fully summarize the similarities and differences among the genotypes, for the purpose of considering the state of development of the field in terms of its theoretical basis, it is useful to review what each genotype has to say about these three views of work. Figure 2.9 shows this contrast.

The CMU genotype approximates work as a goal-driven task. As a result, all aspects of the work domain and agent cognitive processes considered are embedded in the task view (i.e., GOMS family of models). Conversely, the OSU genotype stresses interactions and a demand characteristic perspective of work. Thus, the agent and task are embedded in the work domain framework. The RISO genotype is also organized around the work domain, but it provides separate frameworks that may be regarded as relating more directly to the other views of work. The same may be said for the UCSD genotype, if Norman's cognitive control framework from cognitive psychology is included as a component. But in this case, the domain view is much more restricted, consisting of the human-computer interaction and a purpose-driven task.

The genotypes also differ in the degree to which they have refined their respective frameworks. The CMU framework provides a process architecture based on a transformation paradigm. Abstract, purpose-driven goals are transformed by a sequence of operators to subgoals and eventually into elemental actions. The RISO genotype is the most developed framework from the ecological perspective. Separate, interconnected sub-frameworks provide detailed coverage of all three views of work. They may be considered as containing an emergent process architecture from a self-organizing control perspective. The architecture is necessarily represented implicitly rather than explicitly. The explicit representation identifies factors that shape the emergent process, including intended and unintended driving forces embodied in the action agent.

The OSU and UCSD conceptual frameworks are not as thoroughly developed as the other two. The OSU orientation is consistent with a self-organizing notion of work and thus one might expect it to develop along the lines of RISO genotype. The UCSD genotype seems to provide a mixed case. It is formed from a traditional cognitivist perspective that emphasizes internal human information processing. The seven-stage model reflects this transformational process. However, it also gives recognition to experiential based behavior that involves attuning to the environment, which seems to suggest the importance of a self-organizing system being implicitly shaped by external factors. But it is not at all clear how these two different types of control processes interrelate to define the texture of work.

Conceptual Views of Work			
Genotype	Agent View	Task View	Domain View
CMU	MHP model	GOMS family of model	
OSU			Cognitive triad framework
RISO	SRK framework	Control task framework	Abstraction Decomposition Space framework
UCSD	Experiential and reflective control frameworks[9]	Seven-stage model	Gulfs of execution and evaluation; direct manipulation

Figure 2.9: Human work conceptualized from three different perspectives: An agent view, a task view, or a domain view. This chart shows the relation between these perspectives and the conceptual tools used by each of the four CSE genotypes.

Each genotype might be regarded as providing a different hypothesis about how to model work in a way useful for design engineering. The ecological stance of the OSU and RISO genotypes argues the position that work is emergent and should be modeled indirectly. The indirect approach is preferred because the model can be robust to change and supports dynamic error correction. These attributes are achieved in part by explicitly representing the coupling between *task activity* and the *work domain*. As disturbances arise, knowledge of the domain can be used to make adaptive adjustments in the work pattern. In other words, *work* emerges from the interactions of a work field, and a task relative to an agent or worker. Thus, the ecological position emphasizes the capture and representation of adaptive behavior in an open fashion. It is from this basis that the constructs to model work were selected.

The CMU and UCSD genotypes have elected to model work in a *direct, single goal-driven manner,* whereas the ecological approach, at least implicitly, considers multiple goals. An advantage of this approach is that work can be modeled as a closed process. As a result, it provides a means to more easily establish predictions of human performance based on the consequences of cognitive work. In general, one would expect to be able to make human work models faster from this perspective. It readily accommodates multiple levels of approximation that can be selected to meet the system development needs as they arise. However, this gain is offset by a reduction in the ability of the models to capture adaptive behavior in a robust manner, and it does not address error recovery unless it is explicitly built into a model. The indirect approach may also generate different levels of approximation, but because of the diversities of vari-

9. These control processes reflect Norman's view of data-driven and cognitive-driven behavior. They have been developed as part of Cognitive Psychology and represent an expression of the general notion that there are two modes of mental control, so-called automatic and control processes. It is not clear if this property of a cognitive model is formally included in the UCSD CSE genotype. It was not discussed in the overview presented here.

ables involved and the limitations of less-developed modeling tools for self-organizing, nonlinear dynamic systems, it is more difficult to cast these models in a computational form that can support performance prediction.

In summary, CSE has evolved a complex conceptual landscape. Work in this field is advancing the general scientific knowledge of human work. By nature, cognitive variables are ephemeral. They are of necessity abstractions. We cannot directly see or touch them. As a result they are open to a great deal of controversy, and they are difficult to define with precision. These four CSE genotypes provide a flavor of the range of variables that are believed to be useful for characterizing work to aid design engineering. They suggest that cognition resides both in the head and in the environment based on the characteristics of objects relative to an agent's work and knowledge state. While the conceptual distinctions allow us to discriminate among these different genotypes, it is fair to say more work is needed to improve the definition of terms and the creation of cognitively based work architectures to model human work.

2.4 CSE ENGINEERING PRACTICES

The groundwork for cognitively oriented engineering analysis was largely established during the 1980s. This provided basic conceptual tools to be used to support the human-centered design of interactive systems from the perspective of each of the genotypes. The concepts of work and approaches to modeling work address the subject matter of the field. The next issue is how to gather and analyze actual information about specific work to be used to populate the perspective frameworks to support design decision making from a human work perspective. Each genotype places its own emphasis on what to look for to characterize cognitive work. Each one has made its own contributions to the analysis, design, and evaluation aspects of systems engineering. The practice of CSE also has been maturing in each of these aspects of design and development. These developments are discussed in this section. For the purpose of organization, I will consider these developments separately in terms of support to the analysis, design, and evaluation aspects of systems development.

2.4.1 Analysis

CSE is most closely associated with human-centered analysis of an interactive system. Indeed, some seem to consider cognitive systems engineering as being the analysis activity that has come to be called Cognitive Task Analysis (CTA). There is more to CSE than CTA, but it should be obvious that CTA is heavily involved in the process of populating any of the previously discussed CSE frameworks to model work in a way useful for design engineering.

The origins of task analysis date back to the beginning of human factors engineering as an approach to user interface design. It includes a set of techniques to assess and describe human-machine interactions systematically relative to work, usually expressed as a task. Like all other aspects of systems engineering, a structured analytical process is useful to help ensure that all relevant issues have been identified and considered in the design, and to provide a rationale for identified requirements. Further, it is also the case that a systematic and structured process serves to lend credence to any judgments that must be made in the analysis. These same benefits apply equally as well to CTA.

CTA is mainly different from classical task analysis in two ways. First, as expected, it attempts to make explicit the cognitive aspects of task work. Second, for some forms of CSE, it includes a formal representation of the task or work domain, as well as work activities.

In broad terms, a CTA, as well as a behavioral task analysis, includes the use of techniques to extract and derive information about work that can be used to establish system requirements. Document analysis and various forms of knowledge elicitation techniques, including nonintrusive observation studies, are widely used as a means to gain deeper insights into work in general, and cognitive processes in work in particular. These methods are used by practitioners who approach the analytic work of systems engineering from one of the four CSE genotypes, as well as by those who may take a more eclectic CSE approach, and other human factors engineers, human-computer interface specialists, and systems analysts who do not identify with the CSE movement. However, even though the same basic *forms of data collection* are common, each practitioner will have preferences that influence how they probe, what they look for, and the judgments they make in the process of extracting information from the raw data. For example, a GOMS-oriented analyst will use document search techniques, interactions with subject-matter experts, and observational techniques to establish a functional, single goal-decomposition model of work, and use judgment to postulate a work process and level of expression granularity to include in a GOMS model of work. Obviously, the analyst will be primed to notice factors that suggest operators, methods, and selection rules. Other "incidental" factors may be missed, treated as noise, or considered to be second or third order issues, and thus they will not influence the GOMS representation. By contrast, a RISO-oriented analyst may be primed to consider some of these same observed "low priority" items as central shaping factors that need further investigation to understand them more fully. Thus, the CTA would probe in these areas. For example, the larger array of factors that serves to establish "priority values" in the work domain fall outside of the factors that directly define functions of a task. The GOMS-oriented analyst would tend to ignore or place less value on these factors, while the RISO-oriented analyst would try to capture the full set of domain values and learn about their interrelationships. Cognitive work, then, would be taken to be

involved as much with managing how these values are satisfied as it would be with operating on the more direct functional and goal-driven properties of work. This aspect of cognition would not be reflected in the GOMS model.

In summary, cognitive engineers and other human factors and human computer interaction specialists may use the same CTA techniques and methods. However, what they extract and how they use the available information is likely to vary widely depending on their orientation and skill.

Our main concern here is to inspect the progress that has been made in the development and use of CTA tools to improve the analysis of cognitive factors in work. Advances in CSE can be seen in three areas. First, more attention is being given to the development of complete systems of analysis, rather than the use of a disconnected set of techniques. Second, new methods have been developed that specifically probe for different aspects of work knowledge and that can be used to represent the qualitative data derived from elicitation and observational techniques. Third, computer support tools continue to be developed to support analysis generation, results sharing, and reusability of work models across projects and for the life cycle of a single project. Some examples of these advances are provided to give a flavor of the state of the art.

2.4.1.1 Systems of CTA. It should be obvious that the four CSE genotypes may be regarded as systems for CTA. Each provides a guiding framework for what knowledge needs to be gained from the analysis of specific work related to a specific development effort. Each framework provides a focus for what to notice, what to probe for, and what the analyst needs to understand. They provide the starting point for a systematic and thorough data collection plan. As indicated earlier, the CMU and RISO genotypes were the most developed frameworks in the 1980s. The OSU genotype has added a Functional Abstraction Hierarchy (FAH) to its framework as an adjunct to the cognitive triad. Other frameworks have also emerged from researchers whose origins are in the decision science community. These include the work on naturalistic decision making championed by Klein (1989; Klein et al., 1993; 1989), and cognitive analysis in support of the development of decision support systems originated by Zachary (1986). This latter work will be discussed in the next section. Here I will update the OSU genotype with a very brief discussion of the FAH.

Both the OSU and RISO genotypes approach cognitive task analysis from an ecological orientation. It is not surprising, then, to find some degree of convergence between these frameworks in terms of the tools used to represent the work domain or the field of practice. The RISO genotype has made a clear distinction between modeling the work domain and modeling work activities that occur in a domain (e.g., Rasmussen et al., 1994; Vicente, 1999). The Abstraction Decomposition Space (ADS) is one of the framework tools used by the RISO genotype to focus and organize a cognitive analysis with respect

to the work domain. More recently, the OSU genotype has developed an FAH as an engineering modeling tool for the same purpose. The abstraction dimension in this approach is based on Rasmussen's AH which is part of the ADS. What is unique about the FAH is that it also represents decision activities as part of the domain (Roth & Mumaw, 1995). Thus, it is used to represent knowledge gained from a cognitive task analysis in the form of a domain model that includes a description of the field of practice in relation to decision-making activity. Whereas the RISO genotype uses separate modeling tools to express the work domain and task activities in decision terms, the OSU genotype brings these two aspects of the CTA data together within a single FAH representation. The inclusion of decision processes in the FAH reflects the OSU design focus on the development of decision support systems.

2.4.1.2 Knowledge Elicitation and Representation. Many new knowledge-gathering techniques have been introduced to better reveal cognitive aspects of work. Klein and his colleagues have been particularly active in this area. They have developed the Critical Decision Method (CDM) and the Applied Cognitive Task Analysis (ACTA) set of methods. The CDM uses a retrospective strategy organized around critical incidents. A Subject-Matter Expert (SME) is asked to recall a specific hard problem or difficult incident, and then an interviewing process is used to issue cognitive probes to identify key decision points, shifts in the situation, and work strategies used. Importantly, eleven specific probe categories are used to acquire a detailed picture of the factors that influenced decision making (Hoffman, Crandall, & Shadbolt; 1998; Klein et al., 1989). The analyst uses probes to encourage the SME to talk about work in terms of analogies, the basis of or "whys" of choice, felt (perceived) time pressure, situation assessment, and hypothetical thinking.

ACTA is a collection of three techniques to capture the cognitive demanding aspects of a task (Militello et al., 1997). ACTA has the virtue that it is easy to learn and apply. The three techniques are called task diagram, knowledge audit, and simulation interview. The task diagram method asks an SME to break down a task into a set of three to six subtasks that are most "cognitively challenging." Using an interactive procedure between the analyst and SME, these subtasks are represented in a flow diagram. The knowledge audit is another probe technique. Importantly, it is organized around probe categories that are known to be important to detect differences between novice and expert performers. For example, a novice has more trouble communicating a big picture view of a situation than does an expert. The knowledge audit is one of the few techniques that use the construct of expertise as a basis for discovering cognitive details about problem solving and decision-making in a work context. The last ACTA method is the simulation interview. It provides a specific incident and asks the SME to engage in mental simulation of the unfolding sit-

uation. Different probes from the other techniques are employed to help focus and stimulate the process. All three techniques in ACTA are geared toward the discovery, from a cognitive perspective, of how SMEs actually work in their jobs. A detailed account of ACTA is provided in Militello et al., 1997.

In the process of using these and other CTA elicitation methods, the Klein group has also produced some new ways to represent the resulting data. For example, they have used an annotation technique to highlight elicited knowledge contained in interview transcripts resulting from the CDM (Hutton et al., 1997) Other representations include (1) a situation awareness chart to succinctly express how a situation is perceived to evolve; (2) a decision requirements table that highlights difficult judgments and indicates why it requires expertise, types of errors found to occur, and other data; (3) a critical cue inventory that lists the cues used to make a diagnosis; and (4) a cognitive demands table that is similar to the decision requirements chart (See Hutton et al., 1997 for examples).

All these techniques have emerged from a naturalistic decision making framework that has been advocated by Klein (Klein et al., 1989). This framework attempts to understand decision making in context, following the same argument of the OSU genotype. The Klein version of naturalistic decision-making could easily be considered another CSE genotype. It coevolved during the same general timeframe as the other genotypes. The framework is ecologically oriented and related to the OSU genotype in the emphasis it places on the demand characteristics of critical situations. Klein has developed a descriptive model, known as Recognition-Primed Decision Making that accounts for many decisions on the basis of recognizing situation characteristics that guide action taking without the apparent need to explicitly formulate options (Klein, 1989). Because naturalistic decision-making has traditionally been associated with the decision sciences, and because it is reasonably closely related to at least one other CSE genotype, it has not been developed as a separate genotype for the purpose of this commentary.

Another knowledge elicitation technique that focuses on decision-making is called the decision decomposition protocol. The full protocol includes a taxonomy of six decision support techniques and a set of 72 questions that focus on uncovering the goal structure of a decision maker. These questions are arranged into eight separate areas (Zachary, 1986). For example, there are questions on the source of goals (e.g., "What goals, if any, are imposed on the decision maker by higher authority?"), task dynamics (e.g., "Is the decision likely to occur more than once in the current context?"), choice criteria (e.g., "What is the decision maker's stance toward risk taking?"), and several other categories.

The decision decomposition protocol was the initial formulation of a comprehensive system for analyzing and modeling work from a decision-making perspective. The approach has evolved into a system known as COGNET. COGNET stands for COGnition as a NEtwork of Tasks. It is a conceptual

framework that aims to model real-time, multi-tasking, problem-solving work. The COGNET analysis and development process results in the formation of a computational model of problem solving. The architecture achieves a global problem-solving process by employing a blackboard structure that interacts with procedural knowledge that expresses tasks to be achieved in the prevailing situation. As a result, the analyst probes for information that can be used to formulate perceptual knowledge, declarative knowledge, procedural knowledge, and action knowledge. Data is derived from expert decision makers engaged in a realistic problem-solving context. The DDP is used first to establish characteristics of the work situation and explicitly seeks to discover the range of problems that make the work challenging. From this base, scenarios are formed and knowledge is elicited from experts performing the task by using think-aloud and question-answering protocols. Data from the elicitations are analyzed and formed into a representation known as the Decision Task Description language. This language has a well-formed syntax that is similar to that of GOMS. These descriptions are used to create the procedures that are expressed in the full computational model (Zachary et al., 1992; 1998). In sum, COGNET is a fully developed CSE system. It could also be treated as a separate CSE genotype, but again, like Klein's work, it is more closely associated with the decision sciences and for this reason was not identified as a genotype.

Three additional knowledge elicitation methods have emerged from work accomplished by military researchers. One method, called the Constraint Analysis and Synthesis Technique (CAST) considers work to emerge from a shaping process like the RISO genotype. The technique begins with a simulation interview that identifies a specific type of work, a current mission, and a detailed description of capabilities to be provided by a first-of-a-kind weapon system. A scenario is briefly sketched that includes a critical event. The SME fleshes out the scenario. The idea is that they will import their expertise into the situation. A series of probes are then used to uncover the factors that influenced why various attributes were included by the SME in the filled-in scenario. Finally, influence factors are evaluated by the SME in terms of influence perceived strength or fixedness of each factor. The method reveals the constraint profile, the origin of constraints, and their strength and thus identifies how cognitive work is shaped (Eggleston & McCracken, 1987). Strength/fixedness rating is a significant feature of CAST not found in other techniques.

The U.S. Air Force has also explored the use of a participatory design framework as a means that, in part, is used to improve the ability to uncover how experts conceptualize work (McNeese et al., 1995; Zaff, McNeese, & Snyder, 1993). One technique employs concept mapping that begins by providing a SME a core concept and using a nondirective and interactive approach to reveal the concepts that are important to the expert, and hence how they conceptualize work. A particularly challenging problem is how to synthesize a unified cognitive map based on inputs from a set of experts whose individual

maps may vary in terms used and other factors. McNeese and his colleagues have developed a tool to address this issue (Gomes, Lind, & Snyder, 1993).

Another technique that attempts to extract and understand expert cognitive skill is the precursor, action, result, and interpretation method. The technique is unique in that it uses a structured interview that has experts pose problems to other experts as a means to simulate representative work conditions. It attempts to uncover problem solving strategies and other forms of tacit knowledge that contribute to expertise. The verbal commentary resulting from this interchange is captured by an analyst. The analyst also asks probe questions to learn about precursors, interpretations, and other facets of the cognitive work (Hall, Gott, & Pokorny, 1995).

This limited sampling of cognitively based knowledge elicitation and representation techniques serves, in part, to illustrate the problems of how to acquire a useful characterization of work in cognitive terms. As Potter et al. (1998) pointed out, a cognitive task analysis attempts to understand how an expert conceptualizes work. It seeks to uncover:

- Technical terms and meanings used in a work context
- Sensitivities to what items are similar/dissimilar and why, and
- How items are organized (whole-part; cause-effect; spatial; etc.).

Potter et al. adhere to an ecological approach to CSE and subscribe to the OSU genotype. In general, the ecologically inclined practitioners of CSE tend to follow Potter et al. and regard CTA directly from the perspective of user work. Other practitioners that have been motivated by the cognitivist orientation of CSE tend to regard CTA as a means to elicit information about various forms of knowledge used by subject matter experts to accomplish work. In other words, work is seen as the deployment of knowledge in a skillful manner. CSE investigators and practitioners have classified work knowledge in many different ways. These include:

- Action knowledge
- Declarative knowledge
- Perceptual-based knowledge and control
- Procedural knowledge
- Rule-based knowledge
- Skill-based knowledge and control
- Strategic knowledge
- Metacognitive knowledge, and
- Attentional control skill knowledge.

It should be clear that there are several factors that make it hard to acquire information about work from a cognitive perspective. For example, it is gener-

ally regarded that experts have tacit knowledge that in part defines their expertise. The expert is able to demonstrate expertise but is not able to directly explicate all knowledge brought to bear on the issue. As a result, indirect probing and observational methods are needed to discover tacit knowledge. These methods are also needed to avoid the problem that arises when the SME attempts to "help" the analyst by providing what he or she thinks the analyst is looking for. This type of "experimenter demand" is well known in laboratory research performed in psychology. The cognitive analyst may ask, "Do you ever do X?" and the subject matter expert may think, "Well, I could think of it that way" and answer "yes" to the question. But in fact the SME does not consider the issue in that manner, usually for some very good reasons. As a result, the analyst is being inadvertently misinformed because the chosen way to make a probe stimulated an unfortunate demand for an answer. Some SMEs will answer, "Yes, but I do not think of it that way, and here is why." Thus, they provide an opening to overcome the demand of the original question. However, others will only provide the simple but misleading answer. Thus, the analyst must not only attempt to understand the SME's thinking process and skill, but he/she must also take steps to ensure the reliability and validity of the acquired data.

For these reasons the new techniques tend to emphasize observations and more open-ended probes. All the techniques mentioned above provide the cognitive engineer with more well defined and better engineered methods to elicit information about expertise and cognitive work processes. Some of the new cognitively oriented task analysis techniques appear to be more efficient than older methods for uncovering how experts work, especially in high stress, complex work situations. In spite of these gains, however, data collection and analysis remains a major time consuming process, and it has proven difficult to establish good ways to store and retrieve information for later analysis and use. Little attention seems to have been directed toward the issues of inter-rater judgment reliability and accuracy, and how to efficiently import the knowledge gained into models that can be used to further understanding and analysis. While there has been good progress, more is needed.

As stated at the outset, this commentary should not be regarded as a comprehensive review of CTA methods. Recent reviews of CTA methods have been produced by Potter et al. (1998) and Hutton et al. (1997), and the general topic is thoroughly discussed in a recent edited book by Schraagen, Chipman, and Shalin (2000). The Hutton et al. review provides brief descriptions of CTA methods and includes an evaluation of 14 selected methods. The evaluation is focused on the type of expertise that can be extracted from a method. Five evaluation criteria were used: resources required (to deploy the method), experience/skill required (to use the method), purpose of the analysis/method, task complexity, and expertise addressed. Potter et al. reviewed 20 cognitive systems engineering methods. These methods were grouped into six classes: (1) mapping semantic space, (2) functional domain modeling, (3) structured interviews/elici-

tation, (4) functional task modeling, (5) computational modeling methods and, (6) performance/observation methods, and participatory design methods. Each method is described and characterized in terms of four factors, tool support; product/output; relevance to systems design, and key issues. These two documents provide a good overview of the current CTA landscape. Other sources of information on CTA are Kirwan and Ainsworth (1992) and Cooke (1994).

2.4.2 Design

The systems development process is often characterized in terms of analysis, design, and evaluation activities. By nature, development for complex systems is iterative, and these three aspects of the process occur many times before the final product is produced. As a field, CSE provides support to all three aspects of systems engineering from a human work perspective.

Support to design from CSE comes in at least two forms. First, cognitive task analysis, used in conjunction with any of the genotypes or other perhaps less formal frameworks, naturally leads to the generation of functional requirements and specifications. Second, CSE also provides concepts, principles, and guidelines to support more detailed decision-making in the design engineering process. There have been new developments in both of these areas.

In general terms, each CSE genotype makes statements about what work information needs to be made available to a worker. The RISO system selects information items to express the so-called intrinsic characteristics of work for a given work domain. The OSU system focuses on information that defines decision making problems and what is needed (1) to aid the user to visualize or otherwise better understand the current situation, (2) to directly assist in the formulation of a course of action, which may include exploring different alternatives implicitly or explicitly, and (3) to better understand potential risks and unintended consequences of different decisions. In other words, the OSU system focuses on decision support. Both the CMU and UCSD systems consider the information needed to produce a procedure, perhaps with some flexibility, to achieve the goal state of a task. These types of information from all of the genotypes are a direct consequence of cognitive analysis. In this section, we are concerned with how is it conveyed to designers to impact the actual design of the interactive system artifact.

To address this issue, it is useful to first consider the different bases from which artifact design evolves. Design decisions may be characterized as coming from three different sources: analysis, management, and innovation. CTA serves as a work- and user-centered aspect of design by analysis. It reflects the gains derived from planning and using a systematic process. Certain items in any analysis stand out as key design requirements that must be reflected in the to-be-produced artifact. Others are more contentious and often reveal (some-

times difficult) trade-offs between alternative capabilities related to different technologies. Program management methods are used to resolve these issues and thus the to-be-produced artifact also reflects design by management. It is often said that design is a creative endeavor. The designer envisions or otherwise formulates a solution concept based on a general understanding of the design problem. The designer has an insight using tacit knowledge and skills. Design innovation of this nature occurs at the very beginning of a project and reoccurs many times by different designers throughout a project until the design is fully realized in the final medium.

Based on this view of design, contributions to the actual design artifact from CSE are conveyed to the final artifact in at least three ways. First, the CSE analysis methods are used in a systematic manner to provide an obvious case for certain user-centered factors that are persuasive to the project team and project manager on their own merits. Hence, they are included in the design. Second, the presentation of CSE factors may be communicated to senior project management in a manner that gains acceptance on the basis of compelling argument alone. In other words, it wins in a trade-off debate. Third, CSE information may be packaged and delivered to the designer working in the medium of the design artifact in a way that clearly relates CSE requirements to the items of the artifact that are available to be manipulated by the designer. Thus, the cost of inclusion by the hardware or software engineer is sufficiently low so that they are worked into the design without debate.

CTA is used more or less directly to impact artifact design from the design by analysis and design by management perspectives. Recent advances in CSE attempt to influence design by innovation in two ways. First, some effort has been devoted within the OSU genotype, under U.S. Air Force sponsorship, toward the development of a CSE-based design process that produces design artifacts that better link with software design tools and methods. Second, the RISO genotype has developed guidelines for an ecological approach to the design of the user interface for interactive systems and has recently formulated an initial design typology framework as another means of shaping user interface design.

Computer-Aided Cognitive Systems Engineering (CACSE) is the name of a tool that has been developed by the Logica Carnegie Group under Air Force sponsorship. The purpose of CACSE is to more formally integrate cognitive work analysis and cognitive systems engineering methods with the general systems development process (Logica Carnegie Group, 2000; Potter et al. 2000). It is generally recognized that every design organization subscribes to its own design methodology within a systems engineering framework. It may be modified in some ways to meet contractual requirements introduced by a customer. This implies that any design methodology that prescribes a rigid process using a fixed set of tools is unlikely to be adopted by any organization; and even if adopted, it is not likely to survive changes in the organization of procurements over time. Based in part on this belief and in part on the recognition that many

different tools can be used to effectively accomplish a CTA, the CACSE method allows for flexibility in method selections. In its current state of development, CACSE lays out a CTA framework that extends CSE products into actual artifact design and that provides a new descriptive formalism to represent decision requirements to the software engineer.

The general framework is shown in graphic form in Figure 2.10. It attempts to show the growth of understanding that occurs over time as more analyses of the current world are performed and more explorations of the so-called envisioned world (i.e., design concept and the form of the world in which it will operate) expressed by the designed artifact. Like the RISO genotype, CACSE divides knowledge acquisition methods focused on current work into those that attempt to (model) understand the work domain and those that provide understanding about the practitioners and activities in the domain. CTA analyses lead to a level of understanding that serves to produce hypotheses for ways to improve work performance. The envisioned world is explored through the development of prototypes. A major goal of the CACSE process is to produce some CSE artifacts that directly aid the developer in the prototyping process.

The FAH, briefly described earlier, is a major CTA tool in CACSE. A Display Task Description (DTC) template is a new invention of the framework that attempts to provide, in a succinct manner, a constellation of information about a decision requirement. The hope is that this information set will better equip the artifact designer to make good design choices based on a deeper understanding of the support requirement. The template identifies four interrelated categories of information: (1) the critical decisions, (2) aspects of the user who must make the decision, (3) supporting information requirements, and (4) the context in which the decision is required. With this set of information, CACSE provides guidance to the prototype design to facilitate proper interpretation of the decision requirement.

The CACSE framework has inspired the development of a software CACSE tool to serve as an analysis and design support system for system engineering from a CSE perspective. In its current form, CACSE is expressed as a software-based analysis environment that contains an FAH toolkit. It includes drawing, editing, and tracking tools for producing, modifying, and storing an FAH (Logica Carnegie Group, 2000).

A somewhat less ambitious effort has resulted in a similar software tool to aid the development and use of work domain analysis in the RISO genotype. This tool has been produced by Sanderson and her colleagues and is known as the Work Domain Analysis Workbench (WDAW) (Sanderson, Eggleston, Skilton, & Cameron, 1999; Skilton et al., 1998). Like the CACSE tool, it provides drawing, editing, and tracking tools but in this case they implement the features of an abstraction decomposition space representation of a field of practice. It provides several features to aid the analyst in constructing and editing a work domain model. The WDAW has been used on several projects by

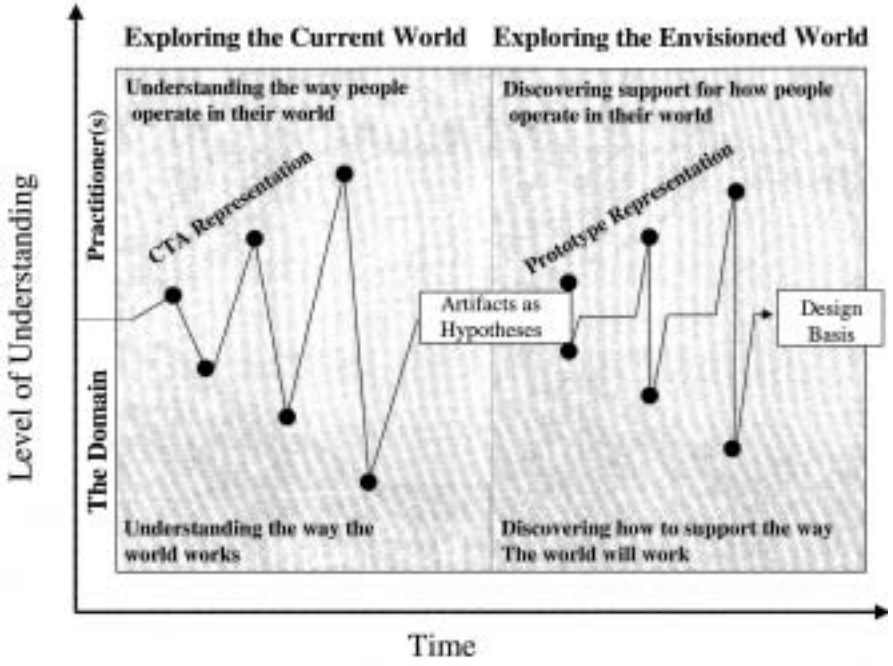

Figure 2.10: The CACSE conceptual representation of cognitive engineering as a human-centered development process that may be integrated with a software engineering development process. (Based on Potter et al., 2000)

Sanderson and her students, and it has recently been employed in a requirements definition study for a major U.S. Navy weapon system.

New computer tools have also been developed to improve support for GOMS modeling. For example, Baumeister et al. (2000) recently compared the use of four different computer aids for constructing GOMS models. These tools varied in ease of use and level of support provided to the analyst. In general, computer tools to support the cognitive analyst tend to reduce the time to produce analysis products, increase the opportunity for data reuse, provide some form of syntactical error checking, and help ensure that a derived work model is well formed with respect to the guiding constructs of a specific CSE genotype. As a result, they help to improve CTA consistency within a genotype.

An interface designer must convert requirements and specifications into a tangible interface artifact. In broad terms the designer must establish a concept for how to represent information, how to present this information to the user, and how to design the interaction with information in the interface. The RISO and UCSD genotypes offer some guidance about how to approach these basic aspects of interface design. The UCSD system embraces the notion of a direct

manipulation interface. By making explicit the characterization of two cognitive gulfs that separate the user from direct work and the need for an explicit first-person engagement with work, the UCSD system provides focusing guidance to the physical artifact designer. This guidance was discussed earlier during the description of the UCSD genotype. The RISO genotype has developed an interface design strategy known as Ecological Interface Design (EID) (Rasmussen, 1988; Vicente, 1999; Vicente & Rasmussen, 1992). EID argues for the merits of focusing on representing the work domain in the user interface, and for providing explicit representation of the domain at both the functional and physical levels of the AH. Further, it points out the need to support work at a skill-based, rule-based, and knowledge-based approach to performance. By providing appropriate abstract information in the presentation, the interface supports reasoning from first principles (knowledge-based), and reveals underlying dynamics that define the state of the system that may be masked when only physical features are presented in the interface expression. In this way, the interface representation and presentation help the user to visualize the work domain state and to recognize disturbances and how to address them. The representation of functional abstraction hierarchy information was shown to improve performance in a recent study by Pawlak and Vicente (1996). It has also been used to prototype the design of new displays for a nuclear power plant facility (Vicente & Tanabe, 1993), and a redesign of the fuel and engines displays for a C–130 Model E–H military aircraft (Dinadis & Vicente, 1999).

In its current form, EID does not provide specific guidance regarding the characteristics of the form of presentation that is best suited to assist visualization of information from each level of the AH. To improve on this state, Rasmussen (1999) has recently proposed a new taxonomy that can be used to identify the representational content for visualization at various levels of the AH. The taxonomy is currently in an early state of development. It attempts to suggest links between the range of representational content by abstraction level and paths to (visualization) presentation to support each type of content.

Given the number of different ways design decisions are made, it is difficult to assess CSE advancement in this area. Cognitive engineers have used their analyses to formulate support concepts (e.g., Potter et al., in press, Roth et al., 2000, Zachary et al., 1992, 1996). CACSE seems like a promising approach to improve the linkage between CSE and the bench-level artifact designer. But in its current state of development it appears to provide more for the CSE analyst than for the software engineer. For example, while CACSE is able to provide the software engineer with design requirements to support worker decision-making, it does not provide any assistance to the engineer with any specific insights about how the requirement relates to the form and content of an effective software object model for the system. More work is needed to narrow the gap and to increase the probability that improved work usage factors will be incorporated into the software design.

2.4.3 Evaluation

Evaluation occurs multiple times throughout the design development process. Requirements, specifications, technology options, design concepts, all are evaluated iteratively as knowledge is accumulated and the project situation changes. Little systematic work has been accomplished toward developing new CSE-oriented evaluation techniques. Rasmussen has sketched a strategy toward a systematic evaluation program that is consistent with the RISO philosophy of design, but it remains in a relatively immature form (Rasmussen et al., 1994). Kirlik and his students have been working on new ways to evaluate data from laboratory-based studies of complex work (Rothrock, 2001; Rothrock & Kirlik, in press).

Recent work by Benda and Sanderson (1998) provides an indication that a different type of evaluation, motivated by the RISO genotype, may gain in importance in the future. Rather than focus on making direct comparisons between alternative concepts, these authors consider the issue of how to evaluate the consequence of some type of change in the work situation. For example, what can be expected if a new piece of technology is introduced into a work domain? Can its impact on total work performance be predicted? Benda and Sanderson point out that many different types of change can map directly into specific modeling frameworks of the RISO system. For example, a technology change that is introduced can be mapped onto the work domain model. They illustrate an evaluation analysis of the impact of technology change by devising a formal notation system that characterizes change in terms of things like tightening and loosening constraints, contradicting current practice, adding affordances, etc. Benda and Sanderson used this notation to evaluate the predicted work domain change resulting from the introduction of a new electronic anaesthesia record-keeping system into a medical center. The basic approach was used to compare how the new technology would influence *coordination* in the domain. It was able to uncover coordination problems that were incidental to the goal-driven value of the specific technology and provided an explanation for why fewer records were signed (a legal requirement) when the new technology was in place than with the older manual recording method. Thus, this form of evaluation considers incidental as well as intentional effects of the introduction of new technology into a work domain.

This type of evaluation is aimed at the scale of the sociotechnical system. It appears to be a promising development that is able to provide systematic evaluation without reducing either the scope or complexity of the work domain in the evaluation process.

Of all the CSE approaches, GOMS modeling from the CMU genotype has received the most use for the purposes of evaluation. Because GOMS models predict outcome performance, they are frequently used to compare alternatives and used at the broad system level (Gray, John, & Atwood, 1993) as well as at

the detailed interface level to compare the details of design concepts. Recently, for example, Gray et al., (2000) has demonstrated that millisecond differences in cognitive processing can add up to produce a significant performance effect associated with interface details like button design. John and Kieras (1996) review several uses of GOMS in this comparative manner.

In summary, GOMS and other process models that capture cognitive factors seem to be useful for usability analysis, including error analysis (Freed et al., 1997; Freed & Shafto, 1997; Gray et al. 2000; John & Kieras, 1996). Both descriptive and computational models are used. The ecological approach to CSE is just beginning to pay more attention to this aspect of the system development process.

2.5 CSE DEPLOYMENT

There is little doubt that the use of CSE approaches has been increasing in the system design community. Examples cited in previous sections provide some evidence to support this claim. Several examples of use have been provided by John and Kieras (1996a), Klinger et al. (1993), Hutton et al. (1997), and Rasmussen et al. (1994) just to name a few sources. In broad terms, CSE has been applied to the information overload problem associated with issues of information superiority in the military (e.g., Flach & Kuperman, 1998; Klein, 1997; Woods, Patterson, & Roth, 1998); the analysis of first-of-a-kind systems (e.g., Flach et al., 1998; Rasmussen, 1998); the analysis and design of decision support systems (e.g., Potter et al., 2000; Roth et al., 2000); and a wide range of work aimed at design trade-off studies and usability analyses (e.g., Gray, John, & Attwood, 1993; John & Kieras, 1996b). This work has resulted in the creation of some novel designs that have been favorably received by both system developers, and it has shown the ability to make nonintuitive predictions of performance that have been validated by empirical investigation (Gray et al., 1993; Roth et al., 2000).

However, it is also important to recognize that there are some factors that may be impeding the rate of progress. These factors range from issues of clarity and ambiguity of use of CSE frameworks, methods, and techniques by CSE practitioners to issues of clarity of understanding of CSE by system designers and managers in the larger system development community. To complete this commentary on the state of the field, it is appropriate to devote some attention to these issues. In this section I will provide examples of these types of problems.

2.5.1 CTA and Other Approaches to Work Analysis

Work or task analysis techniques have been used in human factors and personnel subsystems for many years. Several of these techniques address task description in different forms. (cf. Kirwan & Ainsworth, 1992). Many human factors and Human-Computer Interaction (HCI) practitioners seem to be relatively insensitive to differences in understanding and modeling work tasks when using classical task analytic methods versus the newer, more cognitively oriented methods (Eggleston, 1998). For example, how is a hierarchal task analysis representation (an old technique) different from a cognitively driven, goal-based GOMS representation, or an AH representation of a work domain? Do they capture the same or different information? Do they lead to the same or different conclusions?

It is misguided to place too much emphasis on selective analysis tools. Each CSE genotype, for example, exploits the use of a wide range of techniques for data collection and to support analysis. What is extracted from the techniques differs, sometimes substantially, based on what CSE framework is used. Consistency of use of methods within a framework is probably more important than the specific method selected to meet a given data collection situation. Nevertheless, a mature field is able to make clear distinctions between factors of technical significance. A human factors engineer or human computer interaction practitioner should be able to appreciate the differences in framework and supporting techniques. To illustrate the point, it is instructive to consider the differences, say, in the use of a hierarchical task analysis with the GOMS framework, and the AH framework of the RISO genotype.

Hierarchical Task Analysis (HTA) is an analytic technique used to represent a task in a way that provides a clear and succinct expression of a task useful for design. It represents a task in terms of activities. It produces a hierarchy of a task in terms of operations and plans. Operators identify work requirements for a person in a system. Plans refer to the conditions or rules that specify when to apply an operator. This sounds strikingly similar to the general form of a GOMS model. Indeed, in the Baumeister et al. (2000) study that compared software support tools for GOMS modeling, the example case was originally analyzed by using an HTA. This provides a convenient example to illustrate the often subtle and overlooked differences among the different approaches to capture critical aspects of a task.

The work domain considered in the Baumeister et al. article is a student volunteer scheduling system, named Atropos, used at the annual meeting of CHI'98. Atropos is a software application provided to students to use for self-scheduling of volunteer activities to meet the needs established by the CHI'98 conference planning staff. First, the hierarchical task analysis focused on interaction with the Atropos system. The same is true for the GOMS model. In contrast, the RISO genotype considers work initially from the perspective of the

1.0 Manage SV Schedule
- 1.1 Login
  - 1.1.1 Type User ID
  - 1.1.2 Move Next Field
    - 1.1.2.1 Press <Tab>
    - 1.1.2.2 Home to Mouse
    - 1.1.2.3 Point to Field
    - 1.1.2.4 Click Field
    - 1.1.2.5 Home to Keyboard
  - 1.1.3 Type Password
  - 1.1.4 Press <Enter>
  - 1.1.5 Click<Quit>
- 1.2 Add Slots
- 1.3 View Slots
- 1.4 Delete Slots
- 1.5 Click <Quit>

Figure 2.11: An example of a hierarchal task analysis representation for a volunteer scheduling system. (Based on Baumeister, et al., 2000).

work domain, as opposed to work activity. The AH is a framework that can be used to represent the domain.

The basic structure of the HTA can be represented graphically (see Baumeister et al., 2000) or as an indented outline, where each entry is a goal/operator. For example, the top-level task goal (which is equivalent to a function or an operator) is Manage Student Volunteer Schedule. This is decomposed into five subgoals: Login, Add Slots, View Slots, Delete Slots, and Click <Quit>. The login (function) goal is further decomposed into five lower-order goals, four of which are defined as elemental task activities. The remaining subgoal or operator is decomposed into a set of five elemental tasks. These are presented in outline form in Figure 2.11).

Plan statements are also provided in an HTA. An example plan at the 1.0 level is:

Plan 0: Do 1.1
While not finished, if want to:
Add slot—do 1.2
View slots—do 1.3
Delete slot—do 1.4
Do 1.5

```
Method-for-goal:  Login Atropos
  Step 1.    Recall-LTM-item-whose Content is
    User-info and store-under <User-info>.
  Step 2.    Type-in User-ID of <User-info>.
  Step 3.    Accomplish-goal: Move-to Next-field.
  Step 4.    Type-in Password of <User-info>.
  Step 5.    Keystroke Enter-key.
  Step 6.    Decide:
    If System-response of <User-info> is
      Error-dialog,
   Then Accomplish-goal:
  Recover-from Login-Error-dialog.
  Step 7.    Return-with-goal-accomplished.

Selection-rules-for-goal:  Move-to Next-field
  If User-prefers-KB of  <User-info> is True,
     Then Accomplish-goal : Move-via KB.
  If User-prefers-KB of <User-info> is False,
     Then Accomplish-goal : Move-via Mouse.
Return-with-goal-accomplished.

Method-for-goal : Move-via KB
  Step 1.    Keystroke Tab-key.
  Step 2.    Return-with-goal-accomplished.

Method-for-goal : Move-via Mouse
  Step 1.    Point-to Password-field.
  Step 2.    Click Mouse-button.
  Step 3.    Return-with-goal-accomplished
```

Figure 2.12: A partial GOMS of the Atropos system using the GLEAN 3 GOMS formalism. (Adapted from Baumeister, et al., 2000)

While this HTA appears to be quite detailed, additional work is needed to understand the cognitive consequences of using the system. In other words, if one considers the hierarchal task analysis to be complete, important cognitive aspects of the task will not have been explicitly analyzed and considered in the design. A GOMS model may be used to extend the analysis to provide this additional information. In this way it identifies areas that remain under-specified by the hierarchal task analysis. Some additions from GOMS modeling include (1) forming sets of operators into methods, (2) decomposing operators

into mental operations, and (3) defining cognitive processing times for the elemental operators. An example of one GOMS model for the login operator is shown in Figure 2.12.

This example contains a fragment of a GOMS model that was developed using the GOMS toolkit called GLEAN3 (Kieras et al., 1995). GLEAN3 calculates a time for each execution statement in GOMSL. It adds 50 ms for the execution of every GMOSL statement. In addition it has a time value for each perceptual or motor operator action. For example, 200 ms is added for a mouse click. (These time values are based on studies of human performance.)

In this instance, the CTA accomplished from the GOMS perspective produces a computable model that can be used to predict performance times. It is also diagnostic of areas that may be inefficient from a cognitive load perspective. Thus, in this way the CTA adds additional insights that are not contained in the more traditional hierarchal task analysis method.

CTA often involves a great deal of judgment. Different analysts, for example, might approach the problem differently. An analyst with an ecological orientation, for example, would begin by modeling the student volunteer *work domain*, as opposed to the task activity, by using the Rasmussen AH or the Roth FAH. Figure 2.13 shows a partial, high-level AH for the student volunteer scheduling domain. On the surface this looks similar to the work activity model of the HTA. However, a closer look reveals several differences. First, the functional purpose of the domain acts as the driving goal of the domain and is functionally equivalent to the driving goal treated as the initiator for an activity. Thus, it is the same as the top-level goal used in the HTA and the GOMS model. But the abstract function of the AH codes priority measures that are not expressed in either the HTA or the GOMS representations. Four possible priority measures are identified in Figure 2.13 for the purpose of illustration. These are properties of the domain that influence activities, but are not direct goals of an activity. For this reason they are not captured by the HTA and GOMS approaches to analysis. The ecological view regards these aspects of the environment as important and in the RISO genotype explicitly directs the analyst to look for these features. It is expected that a violation of these values usually results in some undesirable cost that the agents will attempt to avoid, or some positive reward they will seek. These are the nondriving goals that serve to make work complex. Notice also that many of these priority measures go beyond the Atropos software system itself. That is, the work domain from the RISO perspective is larger than the software application, which is where it stops for the HTA and GOMS analysis.

The difference in the AH framework does not stop here. It is also apparent at the general function level. Only one of the items shown at this level deals with the Atropos application directly, make self-appointment option available. The other general functions are ignored by the HTA and GOMS models.

10. The means-end steps represented here are clearly incomplete, leaving out logical steps at the same level of granularity that should be included. However, the abbreviated presentation is sufficient to convey the general structure of a process means-ends, which is the focus of this discussion.

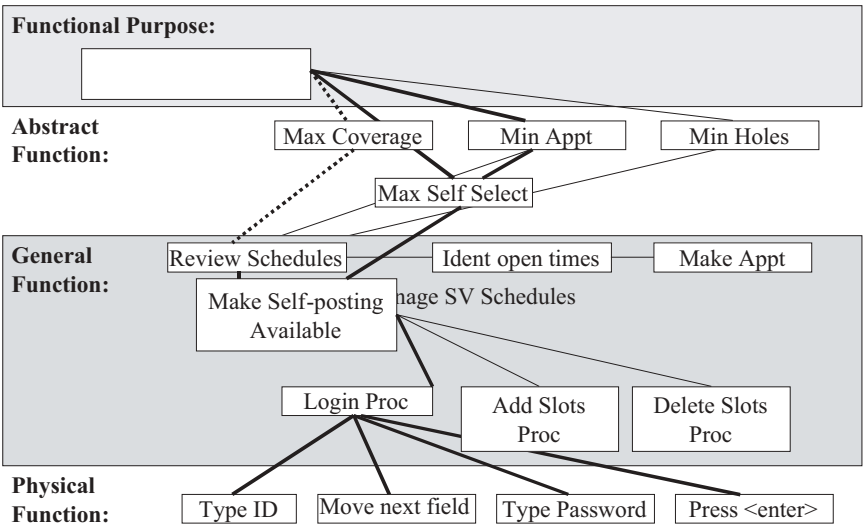

Figure 2.13: An abstraction hierarchy model of the student volunteer scheduling work domain.

For simplicity, only the make self-appointment function is further represented at the Physical Function level. At this point the physical functions are isomorphic with those in the HTA. But remember these only apply to one of the functions shown at the General Function level. Other physical functions would not show up in the HTA.

This brief illustration shows that the information encoded in the AH analysis is organized differently and includes some information covered in the HTA and GOMS, but also includes other information as well. Different conclusions (e.g., requirements) will be drawn from each of these analyses. All of them can provide valuable insights for use by the artifact designer.

2.5.2 Judgment in CSE

To populate any CSE framework requires substantial use of informed judgment by the cognitive engineer. For example, judgment is involved in developing an HTA, a GOMS model, or an AH. In fact, one uses judgment in how to set the system boundary to define the object of study and analysis in the first place. There is no escaping the need for judgment in the analysis of complex problems. However, an analytic process that includes formal analysis within a systematic process helps to reduce subjectivity and contributes to the quality of an analysis. This helps to minimize ad hoc contributions that are both difficult to defend and tend to cut across technical distinctions of different systems or frameworks, thereby adding confusion about the CSE field and its perceived

value. To advance the field, it is important to improve the degree of systemization, guidance, and training provided for each CSE framework.

Proponents of a specific CSE genotype become well versed in a specific brand of cognitive systems engineering. However, it is less clear that CSE developers, students, and practitioners of all forms of CSE fully appreciate and comprehend important distinctions among the different analysis and design approaches. Indeed, there is evidence that some important conceptual distinctions are not always well understood, and this may have a degrading effect on the knowledge gained from a CSE analysis. For example, while it is easy to understand that there is a difference between the work domain and the activities of workers in the domain, similar information about both aspects of a field of practice can be encoded as a property of the *domain*, a property of a *task*, or a property of the *worker*, as indicated earlier (see Figure 2.9). How information is encoded determines what will be made explicit and what will remain implicit in any given representation or model used by a particular genotype of CSE. These differences are important because the user of a representation will tend to be conditioned by the explicit representation and hence tend to miss potentially critical implicit information. Practitioners in a mature field would be sensitive to these differences and be able to evaluate their consequences in relation to different design decisions. Some examples of conceptual distinctions that are often confused by CSE developers and CSE practitioners alike are illustrated below.

2.5.2.1 Confusion Between Process and Structural Means-Ends Analysis. There is evidence that practitioners of CSE tend to be insensitive to technical distinctions promulgated by the different CSE genotypes. One example of this involves the concept of a means-ends analysis that is used both from the cognitivist and the ecological perspectives of CSE.

Vicente (1999) pointed out that the means-end relation coded in an AH is structural in nature and has a different meaning than the means-ends analysis that is consistent with a recursive goal decomposition process expressed in a GOMS model. To make the distinction clear, consider the simple problem of a person feeling dehydrated and setting a goal to solve this problem. In the tradition, a means-ends analysis is a process of recursively decomposing an abstract activity from an abstract goal state until an immediately actionable goal state is produced. The concept was introduced by Simon during the early development of the production system framework (Newell & Simon, 1972; Simon, 1996) and is consistent with the GOMS formalism. For this simple problem, the abstract goal (the end) may be represented as QUENCH_THIRST. A process means-end analysis looks for a way to accomplish this goal. The agent could, for instance, invoke the mental subgoal, GET_DRINK_OF_WATER, which, in turn activates the mental subgoal,

OPEN_REFRIGERATOR, etc., until the activated motor subgoal is a direct action (e.g., POUR_ AND_DRINK).[10] Thus, the means-end hierarchy encodes a process of activity that satisfies the initiating abstract goal.

The process means-end hierarchy may also be used to provide a rationale for an actor's behavior. Any sequence of three subgoals provides an answer to the general questions, why?, what?, and how?, beginning with the most abstract expression. Thus, for example we have:

- WHY: I'm thirsty.
- WHAT: Get glass of water.
- HOW: Get water from the refrigerator.

This same pattern of three questions is used by Rasmussen to highlight a structural means-end relation that maps onto the AH. It can easily be misinterpreted as reflecting a goal decomposition process in the form of a process means-end. It is not. The RISO genotype considers work as an emergent process that is *shaped* by properties of the work environment or domain. These are often called constraints. The emergent work perspective, therefore, is conceived to be a constraint satisfaction process. The constraint objects both serve to explain and *indirectly* cause the observed behavior. Because they operate in this indirect way, it is incorrect to view a structural means-end as goal decomposition, which is a direct transformational process. A structural means-end relation is intended to highlight an indirect shaping process.

To make the point concrete, consider a notional AH for a thirsty person in the context of a typical American home. The system purpose for the domain may be cast as the need to ACQUIRE NUTRITION, as show in the Figure 2.14. That is, the purpose of this domain is established by a physiological state of a human agent. The other levels of the AH are properties of the work domain that further act to shape behavior. For this simple example, the relevant physical form of the domain expresses characteristics of the home environment where the agent is located when a thirsty state occurs.

When an agent (person) is in a thirsty state, the system purpose is activated. This, in turn, serves to activate the other properties of the AH that provide the constraint net that guides the emergent behavior of getting a drink of water from the refrigerator. The process or work activity (of an agent) used to achieve the abstract purpose of the work domain maps results into the same observable outcome behavior as the process means-end, but in this case it is controlled by an indirect shaping process instead of a transformation process. Domain factors shape work activity:

- Acquire Nutrition < shapes need for quenching thirst activity
- Healthy Life Style < shapes choice of thirst quenching activity (e.g., biased toward selection of a low calorie liquid)

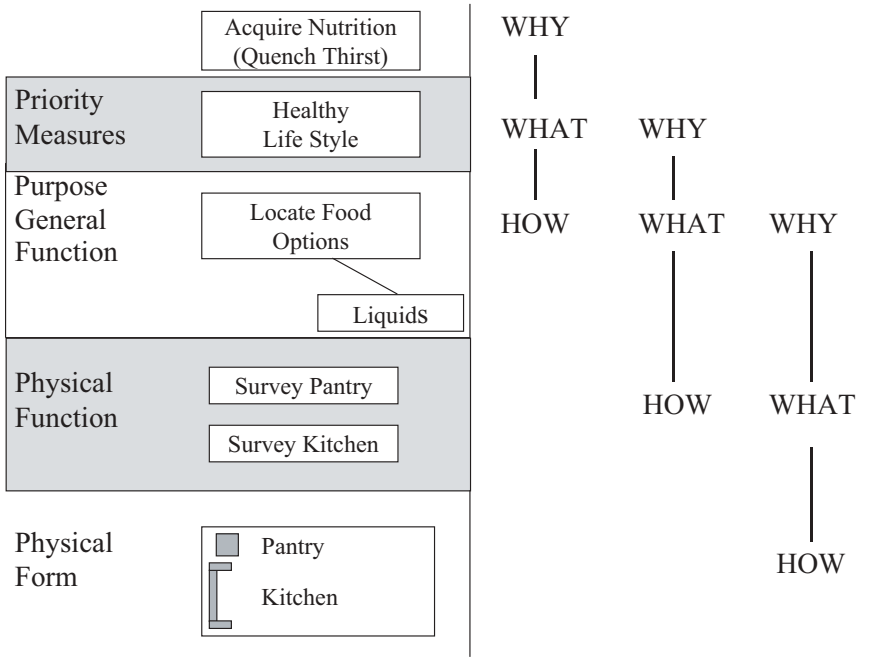

Figure 2.14: A structural means-end relation embedded in an AH representation.

- Locate Food < shapes recognition/reasoning activity (e.g., influences functional looking plan)
- Visually Survey Pantry/Kitchen < shapes action plan activity (e.g., influencing the physical looking plan), and
- Physical Pantry/Kitchen layout < shapes physical action activity (e.g., influences navigate/locomote to refrigerator while avoiding obstacles).

By inspection, it is clear that the AH dimension contains a WHY–WHAT–HOW means-end relation as a way to explain action. ACQUIRE NUTRITION: Why?—I'm thirsty. Because I value a HEALTHY LIFE STYLE, I select a low calorie option (answering the what question). I then recall there is cold water in the refrigerator (functionally answering the how question). In an indirect way, the structural objects of the domain "cause" the behavior of the agent to emerge. The more abstract "ends" are satisfied by the more concrete "means."

In contrast to a process means-end model of goal satisfaction, which establishes a specific trajectory of work, the structural means-end model only shapes outcome behavior and does not specify a given activity path. For example, the domain does not specify the need for water as a subgoal like it would

be in a goal transformation process. Rather, it specifies a constraint on the allowable types of drinks the agent may seek. The agent makes the selection (i.e., closes the degrees of freedom of the problem) as constrained by the shaping factors of the work domain. As a result, the structural means-end relation of the work domain influences the emergent behavioral trajectory while leaving open the actual selections made by the agent.

There are many instances where CSE researchers and practitioners have invoked the Rasmussen AH construct and proceeded to generate a process means-ends model instead of a structural means-end model. This indicates apparent confusion about a major aspect that distinguishes one CSE genotype from another. It is not clear why this happens. Does this mean that this distinction has not been clearly articulated in the literature? Does it mean that researchers and practitioners do not perceive it to be important? Or does it simply mean that there is a strong need for an analyst to create a closed or complete model of behavior instead of creating an indirect model that is incomplete with respect to specifying outcome behavior?

2.5.2.2 Internal Confusion Within the Ecological Perspective on CSE. Even within a given orientation to CSE, there is some evidence that technical distinctions are not well followed and may contribute to creating confusion for CSE students and practitioners. For example, Potter et al. (1998) provide an example of a functional abstraction hierarchy derived from a particular CTA. The illustration uses the functional categories from the Rasmussen AH. But the presentation of the content of the hierarchy creates an impression that is not consistent with the AH concept. Items depicted at the general function and physical function levels reflect whole-part decomposition and do not reflect the abstraction distinction implied by these levels. Further, the hierarchy is identified as an FAH, but it does not reflect any of the unique decision process properties of the FAH. These decision processes may actually be contained in the model and perhaps would be seen if the depiction were expanded, but based on the published data it looks like an example of an incorrectly formed AH, not a FAH. Thus, at best it is misleading and will tend to add confusion for students and users of the CSE product. At worst, it illustrates a muddle between a process means-end and a structural means-end relation within a single representation. What is presented can be interpreted as a mixture of the two types of relations. The authors may be using the AH or the FAH or they have developed an original variant of their own. This creates an opportunity for confusion and misunderstanding both among researchers within CSE who are trying to advance the field and among practitioners who are trying to use the CSE knowledge in systems design.

There is even uncertainty about constructs within a single CSE genotype. One example is suggested by the simple illustration used to show a structural

means-end relation. What are the requirements for defining a work domain in the RISO genotype? Some examples of work domains treat the agent or actor as being external to the domain. For instance, the human body has been taken as a work domain of patient resuscitation (Hajdukiewicz et al., 1999). Other work domains include the actor as a component of the domain. For instance, the medical staff involved in a patient operation is considered as an element in the surgical domain. Does this difference matter? Can a work domain be well formed if it does not include an actor? Under what condition might it be appropriate to define a work domain that does not include an agent? The point here is that this issue and others like it that speak to the precision of a genotype remain as open topics. Until they are addressed, we can expect there will be variations and perhaps inconsistencies in use. This represents another area where additional attention is needed to further advance the CSE field.

Some sources of confusion that reflect the state-of-the-art CSE field are based on an interaction between variations within the field and variations in the larger systems design community. The CSE field has many aims. Some focus on the problem of designing the user-system interface. Others focus on the design of decision support systems. Still others may regard the interactive system interface as a support system; hence it blends the HCI and the DSS aspects into a unified work interface. And some concentrate on the entire interactive system as the design object of CSE. These variations in aim mean some design objects can be in conflict. For example, a good CSE-based design for a stand-alone decision support system many be a poor design from the perspective of the user interface as an integrated, multifaceted support system. CSE recommendations therefore can potentially conflict with other CSE based recommendations. Such conflicts weaken both the creditability and perceived value of cognitive systems engineering by the larger design community.

How a design team is organized also influences how CSE can be applied in the design. For example, large-scale design projects tend to have separate teams that are responsible for the human computer interaction, decision support system, and human factors aspects of the system. As a result, this work organization constrains the possibility of some, perhaps highly desirable, CSE inspired design concepts. The interaction between a design team organization and the views of different CSE analysts may be in conflict. If technical decision makers for the project do not understand the reason for these differences, then they will not have a good basis for preferring one set of CSE requirements and recommendations versus another.

These same factors also contribute to the difficulty of improving the integration of CSE practices and products into the system development process. The situation is even more complicated. Currently there are several different approaches to system development. Some projects follow an incremental building strategy. Others follow a spiral development strategy. Both approaches are used when the required infrastructure for a product must be developed as part

of the product. Recently, more system developments have been for products that take advantage of an "open" infrastructure like the World-Wide Web. When creation of an infrastructure does not need to be part of the to-be-designed product, new variations in the development process are possible. What needs to be done to improve the integration of CSE into system engineering practices? All of the above factors make this a complicated question to answer.

2.6 COGNITIVE SYSTEMS ENGINEERING: WHERE DO WE STAND?

This commentary has touched upon many different factors that speak to the maturity of CSE as a science and as an engineering practice. As an applied science, CSE must deal with understanding work in the context of real-world conditions. There is no doubt that this is a difficult undertaking. The four CSE genotypes express different frameworks that in some sense serve as hypotheses of work. These frameworks are one sign of the progress of the field. The ecological perspective represents a bold move to model work in an indirect manner, as opposed to the more obvious direct manner typified by the cognitivistic perspective. This in itself may be regarded as a significant step forward. Based on the material covered in this chapter, it is clear that scientific progress is greater than this. In fact it is not confined to the nature of work. Both the science and the engineering practice of CSE have resulted in new ideas about how to capture and represent or model information extracted from subject-matter experts. This amounts to a scientific advancement in this applied area as well. The same can be said about the guidance CSE provides to the design of user support and the representation and presentation properties of the user interface of an interactive system.

These gains have made it possible to improve the engineering practices of user- or work-centered design. The inclusion of CSE analysis has produced new insights into human work that have suggested new forms of aiding or support and new forms of interface expression that, at least under some situations, have resulted in improved performance. There is little disagreement that CSE adds value to systems development.

But there are weaknesses in CSE in its present form. Terms used by different genotypes are not always well defined. Formal description languages used by these genotypes range from being fairly well formed to being weakly formed. There are wide variations in what CSE practitioners look for in a CTA, what they extract, and how they model work. The practice of CSE may be characterized as being in a guild stage. Any given practitioner tends to follow the belief and concepts of a selected recognized researcher in the field, and there is little fertilization or even understanding across different perspectives. The field depends more on good mentors than on a good system of education. And relatively little attention has been devoted to issues of reliability and validity of results.

The connection of CSE thinking and products to bench-level artifact designers is still an area that is underdeveloped. Object-oriented design has gained a major position, if not the dominant position, in software development. There have been many attempts to add structure to the software development enterprise. The Unified Modeling Language (UML) is the latest object-oriented framework being used in this capacity. It is a graphic language that can be used to visualize, specify, construct, and document the objects of software systems. While it may be treated as a development methodology, UML is better thought of as a language that can be used within any development system. Thus, while it adds structure it is not overly constraining. This is an important characteristic of UML, and one that should be emulated in a system engineering process that links CSE more effectively with software engineering. The CACSE approach appears to be headed down this path. More work along these lines is needed.

There is little doubt that CSE will continue to advance and increase the potential of positively influencing interactive system design. Based on the information presented here, some steps that can contribute to advancement of the field seem clear. There is a need to move beyond the guild stage. This will require improvements in the education of future CSE practitioners.

There is a need to improve the various frameworks used to both capture cognitively based knowledge from subject-matter experts and deploy it in models of work and human performance. This will require directing more attention to extending and sharpening the constructs used in these frameworks and fostering comparisons in a scientifically accepted manner. Scientific debate fuels progress.

There is a need to look for ways to increase reusability of CSE products and to extend CSE to cover the full life-cycle of a system. If the RISO genotype is right in the belief that intrinsic properties of the work domain can be discovered and represented, then it follows that this type of knowledge should be stable and hence reusable as new technology is available for possible insertion into the domain. Other areas of reusability are also possible. Standard units of cognitive actions, for example, may be established to facilitate activity modeling. Other possibilities for reusability also need to be explored.

CSE has enjoyed great interest and improvement over the past several years. More gains can be expected as we enter the 21st century.

REFERENCES

Baumeister, L.K., John, B.E., & Byrne, M.A. (2000, April). A comparison of tools for building GOMS models. In *Proceedings of the CHI'00, AMC,* 502–509. New York: Computer-Human Interaction.

Benda, P.J., & Sanderson, P.M. (1998). *Towards a dynamical model of adaptation to technological change.* Paper presented at OzCHI 1998, Adelaide, South Australia.

References

Bisantz, A. M., & Vicente, K.J. (1994). Making the abstraction hierarchy concrete. *International Journal of Human-Computer Studies*, 83–117.

Bovair, S., Kieras, D.E., & Polson, P.G. (1988). *The acquisition and performance of text-editing skill: A production system analysis* (Tech. Rep. No. 28). Ann Arbor, MI: University of Michigan.

Broadbent, D.E. (1958). *Perception and communication.* New York: Pergamon.

Card, S., Moran, T., & Newell, A. (1983). *The psychology of computer-human interaction*. Hillsdale, NJ: Erlbaum.

Card, S., Moran, T., & Newell, A. (1986). The model human processor: An engineering model of human performance. In K. Boff, L. Kaufman, & J. Thomas (Eds.), *Handbook of perception and human performance, Vol II. Cognitive Processes and Performance.* New York: Wiley.

Cook, R.I., Woods, D.D., & Howie, M.B. (1992). Unintentional delivery of vasoactive drugs with an electromechanical infusion device. *Journal of Cardiothoracic and Vascular Anesthesia, 6,* 1–7.

Cooke, N.J. (1994). Varieties of Knowledge Elicitation Techniques. *International Journal of Human-Computer Studies, 41*, 801–849.

D'Azzo J.J., & Houpis, C.H. (1966). *Feedback control system analysis and synthesis.* New York: McGraw-Hill.

Dinadis, N., & Vicente, K.J. (1999). Designing functional visualizations for aircraft system status displays. *International Journal of Aviation Psychology, 9*, 241–269.

Eggleston, R.G. (1998). Cognitive systems engineering: The latest fab or a true step forward as an approach to complex multi-person system analysis and design? *RTO Meeting Proceedings 4, Collaborative Crew Performance in Complex Operational Systems*, (Rep. RTO–MP–4) (pp 15–1–15–12). Neuilly-Sur-Seine, France: NATO.

Eggleston, R.G. & McCracken, J.R. (1987). *CAST: A constraint analysis and structuring knowledge elicitation tool*. (Internal Laboratory Report). Wright-Patterson AFB, OH: Aero Medical Research Laboratory

Flach, J.M., Eggleston, R.G., Kuperman, G.G., & Dominguez, C.O. (1998). *SEAD and the UCAV: A preliminary cognitive systems analysis,* (Tech. Rep. WP–TR–1998–0013). Wright-Patterson AFB, OH: U.S. Air Force Research Laboratory.

Flach, J., & Kuperman, G.G. (1998). *Victory by design: War, information, and cognitive systems engineering*, (Tech. Rep. AFRL–HE–WP–TR–1998–0074). Wright-Patterson AFB, OH: U.S. Air Force Research Laboratory.

Freed, M.A., & Shafto, M.G. (1997). Human-system modeling: Some principles and a pragmatic approach. In *Proceedings of the Fourth International Workshop on the Design, Specification, and Verification of Interactive System*, Granada, Spain.

Freed, M.A., Shafto, M.G., & Remington, R.W. (1997). *Employing simulation to evaluate design: The APEX approach.* NASA ASC website.

Gibson, J.J. (1979). *The ecological approach to visual perception*. Boston, MA: Houghton-Mifflin.

Gomes, M.E., Lind, S., & Snyder, D.E. (1993). A human factors evaluation using tools for automated knowledge engineering. In *Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON)*, Dayton, OH. 661–664.

Gray, W.D., John, B.E., & Atwood, M.E. (1993). Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world performance. *Human-Computer Interaction, 8* (3), 237–309.

Hajdukiewicz, J.R., Burns, C.M., Vicente, K.J., & Eggleston, R.G. (1999, September–October) Work domain analysis for intentional systems. In *Proceedings of the 43rd Annual Human Factors and Ergonomics Society meeting,* Houston, TX, 323–327. Santa Monica, CA: HFES.

Hall, E.M., Gott, S.P., & Pokorny, R.A. (1995). A procedural guide to cognitive task analysis: The PARI method; (Tech. Rep. AL/HR–TR–1995–0180). Wright-Patterson AFB, OH: Armstrong Laboratory.

Hoffman, R.R., Crandall, B.W., & Shadbolt, N.R. (1998). A case study in cognitive task analysis methodology: The Critical Decision Method for Elicitation of Expert Knowledge. *Human Factors, 40* (2), 254–276.

Hollnagel, E., & Woods, D.D. (1983). Cognitive systems engineering: New wine in new bottles, *International Journal of Man-Machine Studies, 18*, 583–600.

Hutchins, E.L., Hollan, J.D., & Norman, D.A. (1986). Direct manipulation interfaces. In D.A. Norman & S.W. Draper (Eds.), *User centered system design: New perspectives on human-computer interaction*. Hillsdale, NJ: Erlbaum.

Hutton, R.J.B., Pliske, R.M., Klein, G., Klinger, D., Militello, L., & Miller, T. (1997). *Cognitive engineering implications for information dominance* (Tech. Rep. AL/CF–TR–1997–0172). Wright-Patterson AFB, OH: Air Force Research Laboratory.

John, B.E., & Kieras, D.E. (1996a). The GOMS family of user interface analysis techniques: Comparison and contrast. *AMC Transactions on Computer-Human Interaction, 3* (4), 320–351.

John, B.E., & Kieras, D.E. (1996b). Using GOMS for user interface design and evaluation: Which technique? *AMC Transactions on Computer-Human Interaction, 3* (4), 287–319.

Kieras, D.E. (1988). Toward a practical GOMS model methodology for user interface design. In M. Helander (Ed.), *The handbook of human-computer interaction* (pp. 135–158). Amsterdam: North-Holland.

Kieras, D.E., & Polson, P.G. (1985). An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies, 22*, 365–394.

Kieras, D.E., Wood, S.D., Abotel, K., & Hornof, A. (1995) CLEAN: A computer-based tool for rapid GOMS model usability evaluation of user interface. *Proceedings of the ACM Symposium on User Interface Software and Technology*, 91–100.

Kirwan, B., & Ainsworth, L.K. (1992). *A guide to task analysis.* London: Taylor & Francis.

Klein, G.A. (1997). Implications of naturalistic decision making framework for information dominance (Tech. Rep. AL/CF–TR–1997–0155). Wright-Patterson AFB, OH: U.S. Air Force Research Laboratory.

Klein, G.A. (1989). Recognition-primed decisions. In W. Rouse (Ed.), *Advances in Man-machine systems research, 5*, (pp. 47–92). Greenwich, CT: JAI Press.

Klein, G.A., Calderwood, R., & MacGregor, D. (1989). Critical decision method for eliciting knowledge. *IEEE Transactions on Systems, Man, and Cybernetics, 19* (1), 462–472.

Klein, G.A., Orasanu, J., Calderwood, R., & Zsambok, C. (Eds.). (1993). *Decision making in action: Models and methods*. Norwood, NJ: Ablex.

Klinger, D.W, Androile, S. J., Militello, L.G., Adelman, L., & Klein, G. (1993). *Designing for performance: A cognitive systems engineering approach to modifying an AWACS human computer interface* (Tech. Rep. AL/CF–TR–1993–0093). Wright-Patterson AFB, OH: Armstrong Laboratory.

Logica Carnegie Group (2000, February). *CACSE tool user's manual*, Version 3.0. Pittsburgh, PA: Logica Carnegie Group.

McNeese, M.D., Zaff, B.S., Citeria, M, Brown, C.E., & Whitaker, R.D. (1995). AKADAM: Eliciting user knowledge to support participatory ergonomics. *International Journal of Industrial Ergonomics, 15* (5), 345–364.

Militello, L.G., Hutton, R.J., Pliske, R.M., Knight, B.J., Klein, G. (1997). *Applied cognitive task analysis (ACTA) methodology.* (Final Tech. Rep. Prepared for the Navy Personnel Research and Development Center (contract No N66001–94–C–7034). Fairborn, OH: Klein Associates.

Neisser, U. (1967). *Cognitive psychology.* New York: Appelton-Century-Crofts.

Newell, A., & Simon, H.A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.

Norman, D.A. (1993). *Things that make us smart: Defending human attributes in the age of the machine*. Reading, MA: Addison-Wesley.

Norman, D.A. (1981) *Steps toward a cognitive engineering: System images, system friendliness, mental models* (UCSD Tech. Rep.). La Jolla: University of California Press.

Norman, D.A. (1984) Cognitive engineering principles in the design of human-computer interfaces. In G. Salvendy (Ed.), *Human-computer interaction* (pp. 11–16). Amsterdam: Elsevier.

Norman, D.A. (1986). Cognitive engineering. In D.A. Norman & S.W. Draper (Eds.), *User centered system design: New perspectives on human-computer interaction*.

Norman, D.A. (1987) Cognitive engineering—Cognitive science. In J. Carroll (Ed.), *Interfacing thoughts: Cognitive aspects of human-computer interaction*, (pp. 325–336). Cambridge, MA: MIT Press.

Pawlak. W. S., & Vicente, K. J. (1996). Inducing effective operator control through ecological interface design. *International Journal of Human-Computer Studies*, *44*, 653–688.

Potter, S.S., Roth, E.M., Woods, D.D. & Elm, W. (2000). Bootstrapping multiple converging cognitive task analysis techniques for system design. In J.M.C. Schraagen, S.F. Chipman, & V.L. Shalin, (Eds.) *Cognitive task analysis*. Mahwah, NJ: Erlbaum.

Potter, S.S., Roth, E.M., Woods, D.D., & Elm, W.C. (1998). *Toward the development of a computer-aided cognitive engineering tool to facilitate the development of advanced decision support systems for information warfare domains* (Tech. Rep. AFRL–HE–WP–TR–1998–0004). Wright-Patterson AFB, OH: Air Force Research Laboratory.

Rasmussen, J. (1976). Outlines of a Hybrid Model of the Process Plant Operator. In T.B. Sharidan & G. Johannsen (Eds.), *Monitoring behavior and supervisory control,*(pp 371–383). New York: Plenum.

Rasmussen, J. (1983). Skills, rules, knowledge: Signals, signs, and symbols and other distinctions in human performance models. *IEEE Transactions on Systems, Man, and Cybernetics. 13* (3), 257–267.

Rasmussen, J. (1986). *Information processing and human-machine interaction: An approach to cognitive engineerin*g. New York: Elsevier.

Rasmussen, J. (1988). A cognitive engineering approach to the modeling of decision making and its organization in process control, emergency management, CAD/CAM, office systems, and library systems. *Advances in Man-Machine Systems Research, 4,* 165–243.

Rasmussen, J. (1998). *Ecological interface design for complex systems: An example: SEAD–UAV systems* (Tech. Rep. AFRL–HE–WP–TR–1999–0011). Wright-Patterson AFB, OH: Air Force Research Laboratory.

Rasmussen, J. (1999). Ecological interface design for reliable human-machine systems. *The International Journal of Aviation Psychology, 9* (3), 203–223.

Rasmussen J., Pejtersen, A.M., & Goodstein, L.P. (1994). *Cognitive systems engineering*. New York: Wiley.

Roth, E.M., Gualtieri, J., Easter, & Potter, S.S. (2000). Bridging the gap between cognitive analysis and cognitive engineering. In *Proceedings of the Natural Decision Making Symposium*, Sweden.

Roth, E.M., & Mumaw, R. (1995). Using cognitive task analysis to define human interface requirements for first-of-a-kind systems. *Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting* (pp. 520–524). Santa Monica, CA: HFES.

Rothrock, L. (2001). Using time windows to evaluate operator performance. *International Journal of Cognitive Ergonomics, 5* (1), 1–21.

Rothrock, L., & Kirlik, A. (in press). Inferring rule-based strategies in dynamic judgment tasks. *IEEE Transactions on Systems, Man, and Cybernetics*.

Sanderson, P., Eggleston, R.G., Skilton, W., & Cameron, S. (1999, September-October). Work domain workbench: Supporting cognitive work analysis as a systematic practice. In *Proceedings of the 43rd Annual Human Factors and Ergonomics Society Meeting* (pp 333–337). Santa Monica, CA: HFES.

Schraagen, J.M.C., Chipman, S.F., & Shalin, V.L. (Eds.). (2000). *Cognitive task analysis.* Mahwah, NJ: Erlbaum.

Shannon, C.E., & Weaver, W. (1949). *The mathematical theory of communication.* Urbana, IL: University of Illinois Press.

Simon, H.A., (1996) *The sciences of the artificial* (3rd Ed.). Cambridge, MA: MIT Press.

Skilton, W., Cameron, S., & Sanderson, P.M. (1998). *Supporting cognitive work analysis with the work domain analysis workbench* (WDAW). Paper presented at OzCHI 1998, Adelaide, South Australia.

Vicente, K. (1999). *Cognitive work analysis: Towards safe productive and healthy computer based work.* Hillsdale, NJ: Erlbaum.

Vicente, K., & Rasmussen, J.(1992). Ecological interface design: Theoretical Foundations. *IEEE Transactions on Systems Man and Cybernetics, 22*, 589–606.

Vicente, K.J., & Tanabe, F. (1993). Event-independent assessment of operator information requirements: Providing support for unanticipated events. In *Proceedings of the American Nuclear Society Topical Meeting on Nuclear Plant Instrumentation, Control, and Man-Machine Interface Technologies* (pp. 389–393). LaGrange, IL: ANS.

Wason, P.C., & Johnson-Laird, P.N. (1972). *Psychology of reasoning*. Cambridge MA: Harvard University Press.

Woods, D.D. (1986). Paradigms for intelligent decision support. In E. Hollnagel, G. Mancini, & D.D. Woods (Eds.), *Intelligent decision support in process environments*. New York: Springer-Verlag.

Woods, D.D., & Hollnagel, E. (1983). *Cognitive systems engineering*

Woods, D.D., Johannesen, L.J., Cook, R.I., & Sarter, N.B. (1994). *Behind human error: Cognitive systems, computers, and hindsight*. Wright-Patterson AFB, OH: Crew System Ergonomics Information Analysis Center.

Woods, D.D., Patterson, E.S., & Roth, E.M. (1998). *Can we ever escape from data overload? A cognitive systems diagnosis* (Tech. Rep.AFRL–HE–WP–TR–1999–0195). Wright-Patterson AFB, OH: Air Force Research Laboratory.

Woods, D.D., & Roth, E.M. (1988). Cognitive engineering: Human problem solving with tools. *Human Factors*, *30*, 415–430.

Zachary, WW,. (1986). A cognitive based functional taxonomy of decision support techniques. *Human-Computer Interaction, 2*, 25–63.

Zachary, W.W., Ryder, J.M., Ross, L., & Weiland, M. (1992). Intelligent computer-human interaction in real-time, multi-tasking process control and monitoring systems. In M. Helander & M. Nagamachi (Eds.), *Human factors in design for manufacturability* New York: Taylor & Francis.

Zachary, W.W., Szczepkowski, J. Le Mentec, Schwartz, S., & Bookman, M. A. (1996). An interface agent for retrieval of patient-specific cancer information. In *Proceedings of the Human Factors and Ergonomics Society 40th Annual Meeting*. Santa Monica, CA: HFES.

Zachary, W.W., Ryder, J.M., Hicinbothom, J.H. (1998). Cognitive task analysis and modeling of decision making in complex environments. In J. Cannon-Bowers & E. Salas (Eds.), *Decision making under stress: Implications for individual and team training*. Washington, DC: American Psychological Association.

Zaff, B.S., McNeese, M.D., & Snyder, D.E. (1993). Capturing multiple perspectives: A user centered approach to knowledge and design acquisition. *Knowledge Acquisition, 5*, 79–116.

Zhang, J. (1992). The interaction of internal and external representations in a problem solving task. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp 954–958). Hillsdale, NJ: Erlbaum.